HACK X CRACK: TCP --> COMPRENDIENDO LOS ATAQUES DE INTERNET



PASO a PASO

I.D.S.

SISTEMA DE DETECCION DE INTRUSOS !!!



CONTROLA A TUS ATACANTES !!!

PLUG-INS PARA SNORT

3 SERVIDORES ON LINE PARA TUS PRACTICAS DE HACK

NÚMERO 21

LOS CUADERNOS DE HACK X CRACK

www.hackxcrack.com



CONSTRUYENDO UN PAQUETE TCP EN BINARIO Y DESDE CERO

ESCANEO DE PUERTOS CON SYN

NEMESIS PARA WINDOWS

ENTENDIENDO UN ATAQUE SYN FLOOD

CODIFICACION BINARIA

LOS ESTADOS TCP

SYN FLOOD MEDIANTE HPING2

RAW SOCKETS EN TCP: NEMESIS Y HPING

CURSO DE PHP

* ACCEDIENDO A BASES DE DATOS
* SISTEMAS ODBC Y API
* INSTRUCCIONES SQL
* BASES DE DATOS:
RELACIONALES
ORIENTADA A OBJETOS
EXTENDIDAS

HACK X CRACK - HACK X CRACK - HACK X CRACK

Nº 21 -- P.V.P. 4,5 EUROS



LOS MEJORES ARTÍCULOS GRATIS EN NUESTRA WEB

PC PASO A PASO: PROGRAMACION WEB CON BASES DE DATOS



LOS CUADERNOS DE HACK X CRACK

www.hackxcrack.com

EDITORIAL: EDITOTRANS S.L.
C.I.F: B43675701
PERE MARTELL Nº 20, 2º - 1ª
43001 TARRAGONA (ESPAÑA)

Director Editorial
I. SENTIS
E-mail contacto
director@editotrans.com
Título de la publicación
Los Cuadernos de HACK X CRACK.
Nombre Comercial de la publicación
PC PASO A PASO
Web: www.hackxcrack.com
Dirección: PERE MARTELL Nº 20, 2º - 1ª.
43001 TARRAGONA (ESPAÑA)

INDICE

INDICE DE ANUNCIANTES

Director de la Publicación
J. Sentís

E-mail contacto
director@hackxcrack.com

Diseño gráfico:
J. M. Velasco

E-mail contacto:
grafico@hackxcrack.com

Redactores
AZIMUT, ROTEADO, FASTIC, MORDEA, FAUSTO, ENTROPIC, MEIDOR, HASHIMUIRA, BACKBONE, ZORTEMIUS, AK22, DORKAN, KMORK, MAILA, TITINA, SIMPSIM... ... ...

Contacto redactores
redactores@hackxcrack.com

Colaboradores
Mas de 130 personas: de España, de Brasil, de Argentina, de Francia, de Alemania, de Japón y algún Estadounidense.

E-mail contacto
colaboradores@hackxcrack.com

Imprime
I.G. PRINTONE S.A. Tel 91 808 50 15

DISTRIBUCIÓN:
SGEL, Avda. Valdeparra 29 (Pol. Ind.)
28018 ALCOBENDAS (MADRID)
Tel 91 657 69 00 FAX 91 657 69 28
WEB: www.sgel.es

TELÉFONO DE ATENCIÓN AL CLIENTE: 977 22 45 80
Petición de Números atrasados y Suscripciones (Srta. Genoveva)
HORARIO DE ATENCIÓN: DE 9:30 A 13:30
(LUNES A VIERNES)

NUMERO 21 -- PRINTED IN SPAIN
PERIOCIDAD MENSUAL
Deposito legal: B.26805-2002
Código EAN: 8414090202756

TRABAJANDO CON BASES DE DATOS EN PHP

En este número vamos a manejar la base de datos MySQL en PHP. Conocer como se programan aplicaciones utilizando Bases de Datos es muy importante, ya que nos permitirá, por ejemplo, buscar vulnerabilidades en una Web. Pero todo a su tiempo, primero hay que aprender como va esto de las bases de datos en PHP.

Aplicaciones Web.

Uno de los aspectos más importantes de cualquier lenguaje de programación es el acceso a Bases de Datos (BBDD). Si unimos la utilidad y la potencia de las Bases de Datos a la programación de páginas PHP, podemos programar cualquier aplicación que nos podamos imaginar.

La mayoría de los Websites que existen en Internet, obtienen la información que muestran en sus páginas accediendo a la información existente en una base de datos.

Un claro ejemplo de esto son los portales de noticias. En ellos, la información publicada se obtiene a través de consultas a la Base de Datos (donde se encuentran las noticias). Pero también existen Websites en Internet que insertan, modifican e incluso eliminan datos de una Base de Datos, como por ejemplo una librería on-line, en ella se insertan datos de los clientes, se actualiza el stock de los pedidos realizados...

Arquitectura Web - BBDD

En una Aplicación Web que accede a una base de datos necesitamos los siguientes elementos:

El navegador del cliente.

El servidor de páginas HTML y PHP

El protocolo HTTP que se encarga de comunicar al navegador y al Servidor Web.

La Base de Datos, donde se encuentra la información que se maneja en la aplicación.

Canal de Comunicación: Los mecanismos (ODBC, API de BBDD) necesarios para comunicar al servidor web con la base de datos... ... no te asuste tanta sigla que ahora lo veremos con detalle.

El usuario utiliza el **navegador o browser** tanto para enviar como para recibir las páginas Web. El usuario envía las páginas al Servidor Web y recibe las páginas del Servidor Web a través del protocolo.

El **Servidor Web** se encarga atender las peticiones del usuario (solicitud de páginas), procesarlas y devolverle el resultado. Este Servidor Web será capaz de servir tanto páginas estáticas HTML como páginas dinámicas PHP que se hayan interpretado anteriormente.

La **Base de Datos**, como ya hemos mencionado, es donde se encuentra la información. Hay muchos tipos de Base de Datos y, para el que no tiene ni idea, haremos un paralelismo con los Sistemas Operativos que nos servirá para entender el siguiente punto.

Un Sistema Operativo (por ejemplo Windows) puede mostrarnos el contenido del Disco Duro. En LINUX exactamente igual, es un Sistema Operativo que es capaz de mostrarnos el contenido del Disco Duro. PERO OJO!!! Aunque con ambos Sistemas Operativos obtenemos lo mismo (acceso al contenido del disco duro), la forma en que trabajan internamente es MUY DISTINTA, por lo tanto, el programa que se encarga de acceder y visualizar el contenido del disco duro en Windows es totalmente distinto al de Linux.

En las Bases de Datos es exactamente igual. Hay Bases de Datos de muchos fabricantes distintos (Oracle, SQL, R3...), nosotros podemos tener nuestros datos en Oracle, en SQL o en R3, pero para acceder a ellos necesitaremos instrucciones distintas.

La **Base de Datos** necesita tener un **canal de comunicación** con el Servidor Web encargado de procesar las páginas. Esta comunicación se puede realizar de dos formas diferentes:

ODBC (Open Database Connectivity): permite que una Aplicación Web se conecte a cualquier base de datos que soporte ODBC. Si la base de datos cambia, no hay que cambiar la programación, ya que la forma de acceder a la información por ODBC es la misma para cualquier motor de Base de Datos que soporte ODBC. Como ya vimos antes, si Oracle, SQL y R3 soportasen ODBC, las instrucciones de acceso a los datos serían las mismas (y todos contentos).

API (Application Programming Interface): conjunto de funciones que forman parte de la **interfaz de acceso** a una Base de Datos. Si la base de datos cambia, hay que cambiar la

programación, ya que la API que accede a otra base de datos será diferente. Por ejemplo, las API de MySQL y Oracle son totalmente diferentes.

En muchos casos también es posible utilizar ambas formas de acceso a una base de datos, éste es el caso de MySQL. Si tenemos los drivers necesarios para acceder a MySQL a través de ODBC, podemos utilizar el conjunto de funciones ODBC que incluye PHP. Pero también es posible acceder a esta base de datos a través de su propia API.

Si accedemos a la base de datos a través de la API, la rapidez de acceso será mucho mayor que si accedemos a ella a través de ODBC, ya que ODBC deberá transformar las instrucciones generadas por PHP a otras que entienda MySQL, haciendo por lo tanto más lento el proceso.

Bases de Datos

Las Bases de Datos (sean del "fabricante" que sean) pueden ser:

Bases de Datos Relacionales *(RDBMS, Relation Database Management Systems):* **son las más utilizadas**. Los datos se almacenan en diferentes tablas, que a su vez están compuestas por registros (filas), columnas y campos. Existen una **gran variedad** de BBDD relacionales en el mercado: Access, SQL Server, Oracle, ...

Bases de Datos Orientadas a Objetos *(ODBMS, Object Oriented DBMS):* el dato es representado como un objeto con sus correspondientes propiedades y métodos. Algunas de estas bases de datos son ObjectStore, Versant, GemStone, ...

Bases de Datos Extendidas *(ORDBMS, Object Relational DBMS):* reúne las características de los dos tipos de bases de datos anteriores. Un ejemplo de este tipo de bases de datos es PostgreSQL.

De entre todas las bases de datos que existen en el mercado MySQL forma la pareja perfecta con PHP. Existen versiones de MySQL tanto para Linux / Unix como para Windows, por lo tanto PHP-MySQL se puede utilizar sobre cualquier plataforma.

MySQL es una base de datos ideal para aplicaciones de tamaño pequeño o medio. Su funcionamiento se basa en ejecutar primero un programa residente. Su manejo es a través de la línea de comandos y soporta el lenguaje SQL y conexiones a través de ODBC nivel 0-2.

MySQL almacena cada una de las tablas de la base de datos en un fichero separado. El tamaño máximo de estos ficheros es de 4Gb y la limitación impuesta por el sistema operativo y la capacidad de nuestro disco duro.

En el presente curso trabajaremos con la base de datos MySQL por su integridad con PHP. PHP permite trabajar con las bases de datos más utilizadas, ya sea utilizando su propia librería de funciones (API) o realizando la conexión mediante un enlace ODBC. El soporte de base de datos en PHP incluye algunas como Oracle, MySQL, PostgreSQL, Sybase, ...

Instrucciones SQL

El lenguaje SQL (Structured Query Language) es el lenguaje estándar que se utiliza para manipular la información que contienen las bases de datos relacionales.
No vamos a entrar en profundidad en el lenguaje SQL, ya que esto nos llevaría un tiempo considerable. Pero si que vamos a contemplar los conceptos básicos necesarios para realizar una Aplicación Web.

Seleccionar

Para seleccionar registros en una base de datos utilizaremos la instrucción **SELECT.** Veremos el funcionamiento de la instrucción SELECT a lo largo de los siguientes ejemplos.

Los ejemplos que se van a mostrar van a actuar sobre las siguientes tablas:

Tabla: ALUMNOS

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	21	C1
A2	José	Fuentes	México	47	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3

Tabla: CURSOS

CODIGO	CURSO	HORAS
C1	PHP práctico	180
C2	ASP.NET	150
C3	JSP	150

```
//Selecciona todos los campos de la tabla alumnos
SELECT * FROM ALUMNOS;
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	21	C1
A2	José	Fuentes	México	47	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3

```
//Selecciona el nombre y el apellido de la tabla alumnos
SELECT NOMBRE, APELLIDO FROM ALUMNOS;
```

Donde NOMBRE y APELLIDO son campos de la tabla y van separados por comas (,)
Y el resultado obtenido es

NOMBRE	APELLIDO
María	López
José	Fuentes
Laura	Cardone
David	Márquez

//Selecciona el nombre y el apellido de la tabla alumnos
//cuyo pais sea España

```
SELECT NOMBRE, APELLIDO, PAIS
FROM ALUMNOS
WHERE PAIS = 'España';
```

Donde NOMBRE, APELLIDO y PAIS son campos de la tabla y la opción WHERE filtra por los criterios deseados, en este caso cuyo PAIS de procedencia sea España. Cuando comparemos con una cadena, ésta debe ir rodeada de comillas simples (').

Y el resultado obtenido es:

NOMBRE	APELLIDO	PAIS
María	López	España

Podemos añadir los operadores lógicos AND y OR a la opción WHERE y de esta forma obtendremos filtros con diferentes condiciones.

//Selecciona todos los capos de la tabla alumnos
//que tengan menos de 30 años y cuyo pais sea España

```
SELECT * FROM ALUMNOS
WHERE EDAD < 30 AND PAIS = 'España';
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	21	C1

Ahora vamos a relacionar la tabla ALUMNOS y CURSOS para obtener los alumnos junto al curso que realizan. Para realizar esta relación correctamente las tablas deben tener un campo de relación que en este caso es el campo "CURSO" de la tabla ALUMNOS con el CODIGO de la tabla CURSOS.

//Selecciona el nombre, el apellido de la tabla alumnos
// y el curso de la tabla cursos

```
SELECT ALUMNOS.NOMBRE, ALUMNOS.APELLIDO, CURSOS.CURSO
FROM ALUMNOS, CURSOS
WHERE ALUMNOS.CURSO = CURSOS.CODIGO;
```

Y el resultado obtenido es:

NOMBRE	APELLIDO	CURSO
María	López	PHP práctico
José	Fuentes	PHP práctico
Laura	Cardone	ASP.NET
David	Márquez	JSP

Para ordenar los resultados obtenidos añadiremos al final de la consulta la opción ORDER BY y a continuación los campos por los que queramos ordenar y el sentido de la ordenación ASC (ascendente) y DESC (descendiente).

//Selecciona todos los capos de la tabla alumnos
//cuyo curso sea el C1 y los ordena por su edad

```
SELECT * FROM ALUMNOS
WHERE CURSO = 'C1' ORDER BY EDAD ASC;
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	21	C1
A2	José	Fuentes	México	47	C1

Utilizando la opción **LIKE** en vez del operador **=** podremos comparar parte de una cadena.

//Selecciona todos los capos de la tabla alumnos
//cuyo nombre contenga la letra "a"

```
SELECT * FROM ALUMNOS
WHERE NOMBRE LIKE '%a%';
```

//Selecciona todos los capos de la tabla alumnos
//cuyo nombre empiece la cadena "Da"

```
SELECT * FROM ALUMNOS
WHERE NOMBRE LIKE 'Da%';
```

//Selecciona todos los capos de la tabla alumnos
//cuyo nombre termine con la letra "a"

```
SELECT * FROM ALUMNOS
WHERE NOMBRE LIKE '%a';
```

Con LIKE '%cadena%' selecciona registros cuya "cadena" esta contenida dentro del campo a comparar.

Con LIKE 'cadena%' selecciona registros cuya "cadena" empieza por campo a comparar.

Con LIKE '%cadena' selecciona registros cuya "cadena" termina por campo a comparar.

Si sabemos de antemano que la ejecución de una sentencia de selección va a devolver una gran cantidad de registros, se puede limitar esta cantidad de registros.

//Devuelve solo los 10 primeros registros de la selección

```
SELECT * FROM ALUMNOS LIMIT 10;
```

//Devuelve 15 registros a partir de la posición 30

```
SELECT * FROM ALUMNOS LIMIT 30,15;
```

$01 , 07

Actualizar

Para conseguir modificar el contenido de una base de datos utilizaremos la instrucción **UPDATE**. Comprenderemos mejor su funcionamiento a través de los siguientes ejemplos.

Si queremos modificar el contenido de un campo de un registro en concreto utilizaremos la instrucción UPDATE seguida de la opción WHERE para filtrar el registro que queremos modificar.

```
//Modifica el contenido de la edad de María de la tabla alumnos
UPDATE ALUMNOS SET EDAD = 25 WHERE CODIGO = 'A1';
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	25	C1
A2	José	Fuentes	México	47	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3

Si queremos modificar más de un campo del mismo registro, añadiremos tantos *campo=valor* separados por comas, como campos queramos modificar.

```
//Modifica el contenido de la edad de María de la tabla alumnos
UPDATE ALUMNOS SET EDAD = 25, CURSO = 'C2'
WHERE CODIGO = 'A1';
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	25	C2
A2	José	Fuentes	México	47	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3

Si queremos modificar más de un registro al mismo tiempo en la opción WHERE deberemos aplicar un filtro que afecte a más de un registro.

```
//Modifica el contenido de la edad de María
//de la tabla alumnos
UPDATE ALUMNOS SET EDAD = 25
WHERE CURSO = 'C1';
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	25	C1
A2	José	Fuentes	México	25	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3

Si queremos modificar todos los registros de una tabla, simplemente no deberemos añadir la opción WHERE a la instrucción UPDATE.

```
//Modifica el contenido de la edad de María de la tabla alumnos
UPDATE ALUMNOS SET EDAD = 25
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	25	C1
A2	José	Fuentes	México	25	C1
A3	Laura	Cardone	Argentina	25	C2
A4	David	Márquez	Venezuela	25	C3

Insertar

Para añadir o insertar nueva información en una base de datos utilizaremos la instrucción **INSERT**. Veremos el funcionamiento de INSERT a través de los siguientes ejemplos.

```
//Añade un registro a la tabla alumnos
INSERT INTO ALUMNOS
(CODIGO, NOMBRE, APELLIDO, PAIS, EDAD, CURSO)
VALUES ('A5', 'Sabina', 'Campoy', 'España', 17, 'C1');
```

Y el resultado obtenido es:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1	María	López	España	21	C1
A2	José	Fuentes	México	47	C1
A3	Laura	Cardone	Argentina	33	C2
A4	David	Márquez	Venezuela	27	C3
A5	Sabina	Campoy	España	17	C1

Es importante recordar el tipo de dato que puede almacenar cada campo. Las inserciones de datos tipo *string* deben ir entre comillas, en cambio los de tipo numérico no.

En el uso de INSERT no es necesario especificar ningún filtro (WHERE), ya que la inserción no afecta a ningún registro en particular, si no que afecta a una tabla específica.

Borrar

La instrucción **DELETE** borra un determinado registro de una tabla, para especificar los registros que queremos eliminar lo haremos añadiendo la opción WHERE. Al igual que con la instrucción UPDATE, si no se especifica la opción WHERE, el borrado afectará a todos los registros de la tabla.

```
//Elimina al alumno A2
 DELETE FROM ALUMNOS WHERE CODIGO = 'A2';
//Elimina a los alumnos del curso C1
 DELETE FROM ALUMNOS WHERE CURSO = 'C1';
//Elimina a todos los alumnos
 DELETE FROM ALUMNOS;
```

PHP – BBDD (MySQL)

PHP dispone de un conjunto de funciones que forman la API que permite utilizar la base de datos MySQL. Estas funciones se encuentran optimizadas para sacar el máximo rendimiento a MySQL, por lo tanto serán mucho más rápidas que acceder a MySQL a través de ODBC. La lista de funciones en PHP que manejan el API de MySQL es muy amplia, a continuación veremos las funciones más importantes. El listado completo de todas las funciones lo podemos encontrar en la página web oficial de PHP (www.php.net).

mysql_connect

Abre una conexión a un servidor MySQL

La sintaxis de la función es la siguiente:
int mysql_connect ([string servidor [string :puerto/path_to_socket] [, string usuario [, string password]]])

La función devuelve un valor numérico de tipo *int* que identifica la conexión con el servidor MySQL.
Servidor: nombre o dirección IP del servidor donde está la base de datos.
Puerto: puerto por el que se accede a la base de datos.
Path_to_socket: el socket que el servidor está utilizando para escuchar las peticiones (sólo en Unix)
Usuario: nombre del usuario para realizar la conexión a la base de datos.
Password: clave del usuario para realizar la conexión a la base de datos.

La función *mysql_connect()* establece una conexión a un servidor MySQL. Todos los argumentos son opcionales, y si no se especifican, se asumen los valores por defecto:
servidor: localhost
puerto: 3306
path_to_socket: /tmp/mysql.sock
usuario: usuario propietario del proceso del servidor
password: password vacia.

El servidor también puede incluir un número de puerto (localhost:3306) o un camino al socket (:/camino/al/socket) para localhost.

En el caso de que se haga una llamada a *mysql_connect()* con los mismos argumentos, no se establecerá un nuevo enlace, sino que se devolverá el enlace ya abierto. Esto consigue optimizar recursos, ya que más de un enlace abierto solo consumiría más recursos para obtener los mismos resultados.

El enlace al servidor se cerrará tan pronto como la ejecución del script PHP finalice, es decir cuando termine la ejecución de la página PHP, a menos que se cierre antes explícitamente llamando a *mysql_close()*.

En el siguiente ejemplo se muestra el proceso de conexión a un servidor MySQL que está en la misma máquina (localhost) donde está el Servidor Web. El usuario es "pepe" y la contraseña es "12colina34"

```
mysql_connect("localhost", "pepe", "12colina34");
```

Si el puerto es diferente al puerto por defecto y no es necesario especificar el usuario y password:

```
mysql_connect("localhost:3345", "", "");
```

Lo más práctico a la hora de realizar aplicaciones es declarar la conexión a la base de datos en un fichero php independiente y después incluir este fichero en aquellas páginas que necesiten una conexión a la base de datos.

```
<?php
$servidor = "127.0.0.1";
$usuario = "root";
$clave = "admin";
$miconexion = mysql_connect($servidor, $usuario,
$clave);
?>
```

Si al ejemplo anterior lo llamamos "conexion.php" deberemos incluir el siguiente código en nuestras páginas:

```
<?php
include ("conexion.php");
?>
```

Para poder conectarnos a la base de datos, debemos indicar en los parámetros de la conexión un usuario y contraseña válidos y existentes como usuarios en la base de datos. Recuerda que en el tema anterior aprendimos una administración básica de la MySQL a través de *phpmyadmin*.

mysql_select_db

Selecciona la base de datos con la que se va a trabajar de entre todas las existentes en el servidor.

La sintaxis de la función es la siguiente: *int mysql_select_db (string base_de_datos [, int identificador_de_enlace])*

Devuelve **TRUE** si éxito, **FALSE** si error. Los argumentos de la función son:
Base_de_datos: nombre de la base de datos a seleccionar.
Identificador_conexión: identificador de la conexión.

```
<?php
mysql_select_db("pruebas", $miconexion);
?>
```

La función *mysql_select_db()* establece la base de datos activa que estará asociada con el identificador de conexión especificado. Si no se especifica un identificador de conexión, se asume el ultimoúltimo enlace abierto. Si no hay ningún enlace abierto, la función intentará establecer un enlace como si se llamara a *mysql_connect()*.

Ahora modificaremos el include conexión para añadirle la línea de selección de la Base de Datos donde hemos dado de alta las tablas ALUMNOS y CURSOS.

```
<?php
$servidor = "127.0.0.1";
$usuario = "root";
$clave = "admin";
$miconexion = mysql_connect($servidor, $usuario, $clave);
mysql_select_db("aprende_php", $miconexion);
?>
```

mysql_query

Envía una sentencia SQL a MySQL para que se ejecute

La sintaxis de la función es la siguiente: *int mysql_query (string sentencia [, int identificador_conexión])*

Los argumentos utilizados en esta función son:
Sentencia: la sentencia SQL que será enviada al servidor para su ejecución.
Identificador_conexión: el identificador de la conexión sobre la que el comando de SQL será enviado al servidor de la base de datos.

La función *mysql_query()* envía una sentencia a la base de datos activa en el servidor asociada al identificador de enlace. Si no se especifica un *identificador_conexion*, se asumirá el último enlace abierto. Si no hay ningún enlace abierto, la función intenta establecer un enlace como si se llamara función *mysql_connect()* sin argumentos, y lo utiliza.

La función *mysql_query()* devuelve **TRUE** (no-cero) o **FALSE** para indicar si la sentencia se ha ejecutado correctamente o no. Un valor **TRUE** significa que la sentencia era correcta y pudo ser ejecutada en el servidor. No indica nada sobre el número de fila devueltas. Es perfectamente posible que la sentencia se ejecute correctamente pero que no devuelve ninguna fila.

```
<?php
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
?>
```

Asumiendo que la sentencia tenga éxito, se puede llamar a *mysql_affected_rows()* para saber cuantas filas fueron afectadas (para DELETE, INSERT, REPLACE, o UPDATE). Para las sentencias SELECT, *mysql_query()* devuelve un nuevo identificador de resultado que se puede pasar a *mysql_result()*. Cuando se acabe de utilizar el resultado, se pueden liberar los recursos asociados utilizando *mysql_free_result()*.

mysql_result

Devuelve el dato solicitado de un identificador generado por la sentencia *mysql_query*. La sintaxis de la función es la siguiente: *int mysql_result (int id_consulta, int numero_de_fila [, mixed campo])*

Los argumentos de esta función son:
Id_consulta: es el identificador de la consulta realizada con *mysql_query()*
Numero_de_fila: fila a la que se accede para leer el dato.
Campo: campo de la fila que se quiere obtener.

La función *mysql_result()* devuelve el contenido de una celda de un resultado MySQL. El argumento *campo* puede ser el nombre del campo o tabla.nombre_del_campo. Si el nombre de la columna tiene un alias (campo as campo1),

utilizaremos el alias en lugar del nombre de la columna.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM
ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
$result1 = mysql_result($id_query, 1, 1); // fila 1, columna 1
$result2 = mysql_result($id_query, 2, 1); // fila 2, columna 1
print("result1: " . $result1 . "<br>");
print("result2: " . $result2 . "<br>");
?>
```

Para operar con la función mysql_result() debemos imaginarnos el resultado de la consulta como una matriz:

CODIGO	NOMBRE	APELLIDO	PAIS	EDAD	CURSO
A1 (0,0)	María (0,1)	López (0,2)	España (0,3)	21 (0,4)	C1 (0,5)
A2 (1,0)	José (1,1)	Fuentes (1,2)	México (1,3)	47 (1,4)	C1 (1,5)
A3 (2,0)	Laura (2,1)	Cardone (2,2)	Argentina (2,3)	33 (2,4)	C2 (2,5)
A4 (3,0)	David (3,1)	Márquez (3,2)	Venezuela (3,3)	27 (3,4)	C3 (3,5)

El resultado de ejecutar el ejemplo anterior es:



Cuando se trabaja con un gran resultado, debe considerarse la utilización de una función que devuelva una fila entera ya que estas funciones son MUCHO mas rápidas que *mysql_result()*. Especificando un offset numérico en lugar del nombre del campo, la ejecución será mas rápida.

Las llamadas a *mysql_result()* no deben mezclarse con llamadas a las otras sentencias que trabajan con un identificador de resultado.

mysql_fetch_array

Extrae la fila de resultado como un array asociativo. La sintaxis de la función es la siguiente:
array mysql_fetch_array (int id_resultado [, int tipo_de_resultado])

Los argumentos de la función son:
Id_resultado: identificador de resultado devuelto por *mysql_query()*.
Tipo_de_resultado: constante que indica el tipo de array que devuelve. Puede tomar los valores MYSQL_NUM, MYSQL_ASSOC y MYSQL_BOTH.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
while ($fila = mysql_fetch_array($id_query)) {
  print("Nombre: " . $fila["NOMBRE"] . "<br>");
}
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



Para acceder a los datos se utiliza como índice del array el nombre del campo que lo identifica o bien la posición que ocupa en la selección.

Devuelve un array que corresponde al resultado de la consulta, o FALSE si no quedan más filas.

La función *mysql_fetch_array()* es una versión extendida de *mysql_fetch_row()*. Además de guardar los datos en el índice numérico de la matriz, guarda también los datos en los índices asociativos, usando el nombre de campo como clave.

Si dos o más columnas del resultado tienen el mismo nombre de campo (puede pasar al relacionar tablas), la última columna toma la prioridad. Para acceder a la(s) otra(s) columna(s) con el mismo nombre, se debe especificar el índice numérico o definir un alias para la columna.

La función *mysql_fetch_array()* no es significativamente mas lenta que *mysql_fetch_row()*, sin embargo tiene un valor añadido importante.

mysql_num_rows

Devuelve el número de filas de un resultado La sintaxis de la función es la siguiente:
int mysql_num_rows (int id_resultado)

La función *mysql_num_rows()* devuelve el numero de filas de un identificador de resultado.

En el siguiente ejemplo utilizaremos el número de filas seleccionadas para crear un bucle e ir mostrando el valor del campo "nombre" de todas las filas seleccionadas. Utilizamos la variable del bucle $i para indicar la fila en *mysql_result()*.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
$filas = mysql_num_rows($id_query);
for ($i=0; $i<$filas; $i++) {
  $campo = mysql_result($id_query, $i, "NOMBRE");
  print("campo " . ($i + 1) . ": " . $campo . "<br>");
}
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



Por razones de compatibilidad puede usarse también **mysql_numrows()**.

mysql_field_name

Devuelve el nombre del campo especificado en un resultado La sintaxis de la función es la siguiente:
string mysql_field_name (int id_resultado, int indice_del_campo)

La función *mysql_field_name()* devuelve el nombre del campo especificado. Los argumentos de la función son el identificador de resultado y el índice del campo. Devolverá el nombre del campo asociado.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
$campo = mysql_field_name($id_query, 0);
print("campo: " . $campo . "<br>");
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



Por razones de compatibilidad puede usarse tambientambién **mysql_fieldname()**.

mysql_field_type

Devuelve el tipo de dato del campo especificado en un resultado. La sintaxis de la función es la siguiente:
string mysql_field_type (int id_resultado, int indice_del_campo)

La función *mysql_field_type()* es similar a la función *mysql_field_name()*. Los argumentos son idénticos, pero se devuelve el tipo de campo. El tipo será "int", "real", "string", "blob", u otros detallados en la documentación de MySQL.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOS";
$id_query = mysql_query($consulta, $miconexion);
$tipo = mysql_field_type($id_query, 0);
print("tipo: " . $tipo . "<br>");
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



Por razones de compatibilidad puede usarse también **mysql_fieldtype()**.

mysql_create_db

Crea una nueva base de datos MySQL después de haber realizado una conexión al servidor. La sintaxis de la función es la siguiente:

int mysql_create_db (string base_de_datos [, int identificador_conexion])

Los argumentos utilizados en esta función son:
Base_de_datos: nombre de la base de datos a crear.
Identificador_conexión: identificador de la conexión.

La función *mysql_create_db()* intenta crear una nueva base de datos en el servidor asociado al identificador de conexión.

```
<?php
mysql_create_db("pruebas", $miconexion);
?>
```

Por razones de compatibilidad puede usarse *mysql_createdb()* igualmente.

mysql_drop_db

Borra una base de datos MySQL. La sintaxis de la función es la siguiente:
int mysql_drop_db (string base_de_datos [, int identificador_conexion])

Devuelve **TRUE** si éxito, **FALSE** si error.

Los argumentos de la función son:
Base_de_datos: nombre de la base de datos a borrar.
Identificador_conexión: identificador de la conexión.

```
<?php
mysql_drop_db("pruebas", $miconexion);
?>
```

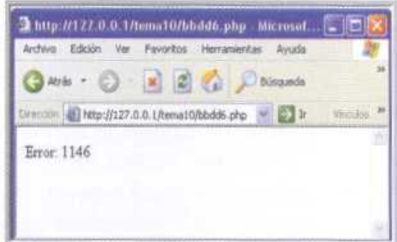
La función *mysql_drop_db()* intenta suprimir una base de datos completa del servidor asociado al identificador de enlace.

mysql_errno

Deuelve el número del mensaje de error de la última operación MySQL. La sintaxis de la función es:
int mysql_errno ([int identificador_conexion])

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOOS";
$id_query = mysql_query($consulta, $miconexion);
print("Error: " . mysql_errno($miconexion) . "<br>");
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



La ejecución del ejemplo anterior generará un error debido a que la tabla ALUMNOS está mal escrita.

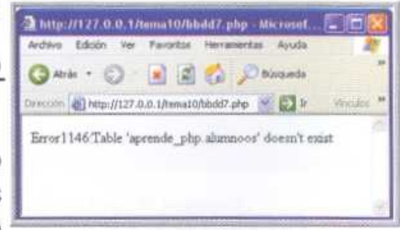
mysql_error

Devuelve el texto del mensaje de error de la última operación MySQL. La sintaxis de la función es la siguiente:
string mysql_error ([int identificador_de_enlace])

Los errores devueltos por mySQL no se indican en los warnings, Por lo tanto debemos usar estas funciones para encontrar el número de error.

```
<?php
include ("conexion.php");
$consulta = "SELECT NOMBRE, APELLIDO FROM ALUMNOOS";
$id_query = mysql_query($consulta, $miconexion);
print("Error".mysql_errno($miconexion).":".mysql_error($miconexion)."<br>");
?>
```

Al ejecutar el código anterior obtenemos el siguiente resultado:



En el próximo número ...

Pues con este número damos por terminado los conceptos básicos de PHP, ahora toca aplicar todo lo aprendido para crear aplicaciones Web seguras. En el próximo número comenzaremos a explicar como programar aplicaciones seguras utilizando PHP.

CURSO DE TCP/IP: 4ª ENTREGA

TCP (TRANSMISION CONTROL PROTOCOL) 2ª PARTE

Este mes nos metemos de lleno en los paquetes de Internet. Crearemos un paquete desde cero, nos adentraremos en el código binario y para rematar comprenderemos de una vez por todas el significado de algunos ataques ya conocidos por todos.

1. Fundamentos de la comunicación digital.

Cuando empecé con el curso de TCP/IP, cuya cuarta entrega tenéis ahora mismo en vuestras manos, ya os advertí de que, cansado de ver una y otra vez las mismas técnicas para explicar los conceptos en todos los libros y tutoriales, este curso pretendía ser una apuesta arriesgada, orientando la explicación de los mismos conceptos desde un punto de vista bastante diferente.

Siguiendo en esta línea un tanto experimental, voy a dar otro nuevo paso que no he visto en ningún libro sobre TCP/IP, para tratar de que os familiaricéis más aún con TCP/IP.

Lo que pretendo conseguir ahora es que convirtáis esos misteriosos paquetes TCP que tenéis rondando ahora mismo por vuestras cabezas, en algo tangible, de carne y hueso, o más bien debería decir de unos y ceros.

Para ello, vamos a ver con un ejemplo cómo está construido exactamente un paquete TCP. Mi intención es que después de este ejemplo, los paquetes TCP dejen de ser al fin para vosotros unos entes teóricos de los cuales conocéis el funcionamiento, pero no su constitución "física".

Por supuesto, esta constitución física no podremos terminar de comprenderla hasta que lleguemos a la capa inferior de la jerarquía de capas de TCP/IP: el nivel físico. Así que habrá que esperar aún unos cuantos meses antes de que veamos qué son exactamente esos ceros y unos de los que vamos a hablar ahora.

De momento, nos quedaremos con una simplificación de la idea de qué es exactamente un cero y un uno. Como sabemos, los datos que circulan en una red lo hacen siempre a través de un medio físico. Este medio, normalmente, será un cable eléctrico, pero puede ser también, por ejemplo, una onda de radiofrecuencia, un cable óptico, o cualquier otro medio físico empleado en la interconexión de máquinas. Quedémonos con el caso más común, el del cable eléctrico, aunque todo lo explicado se puede extrapolar, por supuesto, a cualquier otro medio físico.

Como es lógico, por un cable eléctrico circula electricidad. La electricidad por si misma no contiene mucha información. Un medio de transmisión de información será más versátil cuantos más parámetros posea.

Por ejemplo, una imagen puede transmitir una gran cantidad de información (ya se sabe que una imagen vale más que mil palabras), ya que posee muchísimos parámetros, como son los colores en cada punto, la iluminación, etc. En cambio, la electricidad posee pocos parámetros propios, como pueden ser la tensión eléctrica (voltaje), y la intensidad eléctrica (corriente). Para colmo, estos dos parámetros están directamente relacionados entre sí (¿recordáis la famosa ley de Ohm, o vosotros también suspendíais física en el cole?

Por tanto, una primera solución intuitiva para transmitir información por medio de la

electricidad sería hacer variar esos parámetros en función de lo que quisiéramos transmitir.

Por ejemplo, si queremos transmitir un número, podemos ajustar el voltaje del cable en relación directa con el valor de ese número. Por ejemplo, para transmitir un 5 pondríamos 5 voltios en el cable, y para transmitir un 7 pondríamos 7 voltios en el cable. Si, en cambio, queremos transmitir el número 250000, más nos vale no tocar el "cablecito", a no ser que queramos seguir este curso desde el más allá (allí también llega, desde que la revista cambió de distribuidor).

Supongo que las mentes más despiertas habrán descubierto ya aquí una de las más destructivas técnicas de hacking. Si algún día os encontráis por un chat al <malnacido> ese que os robó la novia, basta con que le enviéis un paquete que contenga un número tocho con ganas, como el 176874375618276543, y por su módem circulará tal tensión eléctrica, que en el lugar que ocupaba su casa veréis un cono atómico que ni el de Hiroshima.

Bueno, antes de que se me eche alguien encima por decir semejantes estupideces, tendré que reconocer que, como es lógico, las cosas no funcionan así en la práctica. ¡Pero no os creáis que sea algo tan descabellado! ¿Y si en lugar de una proporción 1/1 utilizamos otra clase de proporcionalidad para transmitir los números?

Por ejemplo, supongamos que no queremos superar los 10 voltios y, en cambio, queremos transmitir números entre 1 y 1000. Pues basta con que establezcamos el convenio de que 10 voltios equivalen al número 1000 y, por tanto, 5 voltios al 500, 2'5 voltios al 250, etc., etc.

Esta solución no sólo es mucho más realista, si no que incluso ha sido el sistema de transmisión de información en el que se ha basado toda la electrónica hasta que llegó la revolución digital. Y el nombre de esto os sonará mucho, ya que a esto es a lo que se llama comunicación ANALÓGICA.

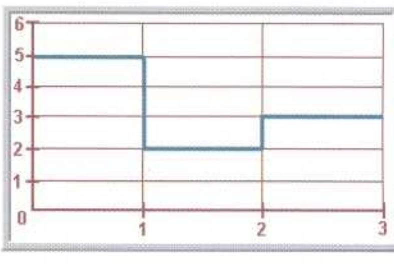
Siempre habréis escuchado el término analógico como opuesto al término digital. Vamos a ver ahora mismo en qué consisten las diferencias entre analógico y digital.

Si bien la tecnología analógica aprovecha la tensión eléctrica, uno de los parámetros que caracterizan la electricidad que circula por un cable, la tecnología digital no utiliza ninguno de los parámetros de la electricidad para transmitir la información.

¿Cómo puede transmitirse la información entonces? Evidentemente, siempre es imprescindible la presencia de una variable que se pueda modificar convenientemente para codificar la información que se desea transmitir. La diferencia es que en el caso de la transmisión digital el parámetro que se utiliza para portar la información no es inherente a la propia electricidad, si no que es un parámetro más sencillo: **el tiempo**.

Evidentemente, el tiempo es siempre un parámetro fundamental en toda comunicación, ya que es imprescindible que haya una sincronización temporal entre transmisor y receptor.

Volviendo al caso de la transmisión analógica, pensemos por ejemplo en el caso en el que se transmitan sólo números del 0 al 9 y, para conseguir representar números más altos lo que se hace es transmitir cada una de sus cifras por separado (unidades, decenas, centenas, etc.). Si, por ejemplo, quisiéramos transmitir el número 523, primero transmitiríamos 5 voltios, luego 2 voltios, y por último 3 voltios.



En la imagen podemos ver la transmisión del número 523 por una línea analógica. En el eje X (el horizontal) se representa el tiempo, por ejemplo, en segundos, y en el eje Y (el vertical) se representa la tensión en voltios.

Lógicamente, es necesario establecer un convenio entre el transmisor y el receptor para saber cuánto tiempo tiene que pasar entre la transmisión de cada cifra. Si no fuese así, imaginad lo que pasaría si tratásemos de transmitir el número 5551. Si la línea se mantiene en 5 voltios el tiempo necesario para transmitir las tres primeras cifras, ¿cómo podrá saber el receptor que en ese tiempo se han transmitido tres cincos, y no sólo uno, o doce?

Por tanto, ha de existir un convenio previo entre emisor y receptor que *diga "cada segundo se transmitirá una nueva cifra"*. Por tanto, si pasado un segundo el voltaje no ha cambiado, significa que la siguiente cifra es igual a la anterior.

Lo que hace la tecnología digital es explotar al máximo este tipo de "convenios". Pero empecemos viendo el caso más simple de todos, que es igual al caso de la transmisión analógica.

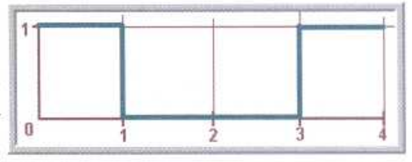
Imaginemos que queremos transmitir números pero que sólo puedan ser 0 o 1. En ese caso, la cosa funcionaría igual: a cada segundo una nueva cifra, y no hay más misterio. La diferencia sería que la tensión sólo podría tener dos valores: 0 o 1 voltios.
La diferencia viene cuando queremos transmitir números más grandes, que es cuando hay que hacer "convenios raros".

En realidad, estos convenios tienen poco de raro, al menos para un matemático, ya que no es ninguna invención, si no simplemente una aplicación directa de las matemáticas. Concretamente, lo que se aplica es la aritmética binaria.

Número decimal	Secuencia binaria
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

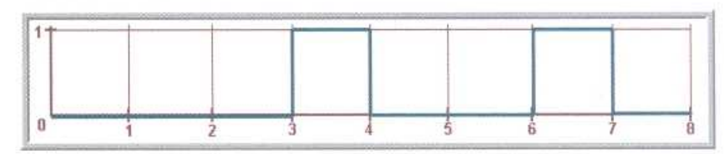
Este "convenio" asigna una secuencia diferente para representar cada número decimal, que podemos ver en la tabla de la izquierda.

Por tanto, si queremos transmitir digitalmente el número 9, estos serán los voltajes que tendremos que poner en la línea:



Es decir, primero 1 voltio durante un segundo, luego 0 voltios durante dos segundos y, por último, 1 voltio durante el último segundo.

¿Cómo podemos entonces transmitir un número de dos cifras, como por ejemplo el 12?
Pues de nuevo hay que hacer otro convenio entre el emisor y el receptor y decir: "cada cifra decimal va a constar de 4 cifras binarias". Por tanto, si transmitimos la secuencia: "0001 0010" (ver la tabla anterior) estaremos transmitiendo el número 12, según el convenio preestablecido, tal y como vemos en la siguiente imagen.



De esta manera, a base de acuerdos entre el transmisor y el receptor, podemos transmitir cualquier información, con sólo transmitir ceros y unos.

¿Cuál es la ventaja de transmitir sólo dos valores en lugar de variar el voltaje en función del valor que se desea transmitir? Pues son varias las ventajas, pero la más obvia es la gran fiabilidad de la transmisión.

Imaginemos que nuestro cable lo hemos comprado en el *Todo a 100* y falla más que una escopeta de feria. Pretendemos transmitir 5 voltios y, en cambio, unas veces el cable transmite 4 voltios, otras veces 3,5, otras veces 2...

Estos fallos del cable serían críticos en una transmisión analógica, ya que el voltaje se traduce directamente en la información que transmitimos. En cambio, la transmisión digital

nos permite unos márgenes de error muy grandes. Al fin y al cabo, transmitir ceros y unos se limita tan sólo a diferenciar *"no hay electricidad en el cable"* de *"sí hay electricidad en el cable"*.

Por tanto, podemos decir por ejemplo: *"si hay menos de un voltio en el cable, consideramos que es un cero. Si hay más de un voltio, consideramos que es un uno"*. Así, todos esos fallos del cable de *Todo a 100* no afectarían a la transmisión digital, ya que incluso 2 voltios seguiría siendo considerado como *"sí hay electricidad en el cable"* y, por tanto, sería considerado como un 1.

Son muchas otras las ventajas de lo digital frente a lo analógico, como la mayor facilidad a la hora de almacenar y procesar la información, pero esas ventajas no se deducen de lo explicado hasta ahora, por lo que no entraremos ahora en más detalle. De momento, la idea con la que nos tenemos que quedar es con que a la hora de transmitir información, la transmisión digital tiene una mayor inmunidad al ruido y, en general, a cualquier error, que la transmisión analógica.

Por supuesto, nada de lo explicado hasta ahora es en la práctica tal y como lo he contado (para aquellos que sepan de qué va el tema y estén flipando), pero mi intención no es escribir un RFC sobre transmisión analógica vs. transmisión digital, si no tan sólo explicar conceptos, aunque tenga que ser faltando a la realidad en muchas ocasiones. Lo importante es que comprendáis los conceptos que subyacen a una transmisión de datos digitales, como es el caso de TCP/IP.

Para los más quisquillosos que sigan insistiendo en que las cosas no son así, y que realmente incluso las transmisiones digitales circulan de forma analógica, insisto en que no estoy tratando de explicar el nivel físico (cosa que ya haré dentro de unos cuantos artículos, y donde todo esto quedará finalmente aclarado), si no tan sólo abstraerme de los detalles para explicar los conceptos básicos.



¿Y si el mundo...

¿Y si el mundo fuese distinto?
Muchas personas, cuando les dices que el mundo informático funciona con CEROS y UNOS, nunca llegan a entenderlo. Con lo sencillo que sería trabajar con el sistema decimal (0,1,2,3,4,5,6,7,8,9) e incluso con letras.

Acabamos de descubrir que aplicar el sistema decimal supondría 10 niveles de voltaje en un cable eléctrico, algo técnicamente posible pero MUY CARO. Como ya hemos dicho los cables deberían ser muy buenos y los dispositivos que detectasen los cambios de voltaje deberían ser muy precisos. Pero esto es hablando del mundo analógico... si nos vamos al digital la cosa se pone interesante.

Hemos dicho que en el MUNDO DIGITAL solo hay dos estados, es decir, "unos" (abierto, con electricidad, iluminado) y ceros (cerrado, sin electricidad, apagado). Pero ¿por qué? ¿por qué el hombre ha decidido que el mundo digital solo tenga dos estados en lugar de 10?

Muy sencillo, POR DINERO!!! ¿Cómo? ¿qué?... ... En el diminuto universo que conocemos, nuestro planeta, es muy sencillo (y barato) encontrar sustancias que sean capaces de tener dos estados claramente diferenciados, que puedan ser capaces de pasar de un estado a otro muy rápidamente, que lo hagan de forma barata y para colmo que sea muy fácil detectar esos estados.

Al igual que una bombilla puede estar encendida o apagada y todos podemos percibirlo mirándola (luz/oscuridad) o tocándola (calor/frío), en el caso de la informática EL DIOS ES EL SILICIO. Simplificando mucho el tema, podemos decir el silicio deja pasar la electricidad o no (ceros y unos), lo hace rapidísimamente (todos queremos tener un PC ultrarrápido) y detectar el estado ("cargado" o "descargado") es tecnológicamente sencillo y por lo tanto barato.

¿Y si el mundo fuese distinto?
Imagina otro elemento, por ejemplo el agua. Todos conocemos tres de sus estados, el sólido, el líquido y el gaseoso... si el agua se pareciese al silicio (si fuese sencillo pasar de un estado a otro, lo hiciese a la velocidad del rayo y fuese sencillo detectar esos cambios de estado)... nuestro ordenador estaría basado en un procesador de agua y, en ese caso... sería MUCHO más potente y rápido que los que ahora tenemos basados en silicio.

¿Qué? ¿Cómo? Sí, porque tendríamos un sistema de TRES estados (apagado -hielo-, neutro -líquido- y encendido -gaseoso-)

en lugar de DOS estados (apagado/encendido). Eso significa que en lugar del código BINARIO utilizaríamos el código TRINÁRIO (el nombre me lo acabo de inventar). Al haber tres estados en lugar de dos podríamos crear convenios mucho más optimizados, es decir, transmitir información de forma mucho más optimizada y por lo tanto mucho más rápida.

Deja volar tu imaginación... si un buen día alguien encuentra (o fabrica, o trae de otra galaxia) un material parecido al silicio pero que pudiese tener 600 estados en lugar de dos... bufff... el mundo informático daría un salto astronómico en velocidad de cálculo. Para la humanidad sería comparable al paso de la edad de Piedra a la edad de Hierro.

Actualmente, a falta de materiales nuevos los científicos intentan utilizar elementos conocidos pero difíciles de "controlar". Unos basados en dos estados y otros en multiestados... por ejemplo los átomos.

Al final, la orgullosa humanidad depende de los materiales que su "humilde" entorno proporciona.

2.- Codificación binaria.

Llegados a este punto, posiblemente ya habréis perdido un poco de miedo a eso de los ceros y los unos. Lo que nos queda por ver es cómo se codifica realmente la información en binario (utilizando tan sólo ceros y unos), es decir, cuáles son los "convenios" reales de los que hemos hablado, que permiten que un transmisor y un receptor se puedan entender.

Es aquí donde entra el concepto fundamental de **palabra (word).** La forma en que se representa la información depende del tamaño de la palabra utilizada. En el ejemplo anterior, donde representábamos cada número decimal con 4 cifras binarias, es decir, con 4 **bits** (bit es simplemente el nombre dado a una cifra binaria, es decir, un número que sólo puede ser un cero o un uno), teníamos una **palabra de 4 bits** para representar los números decimales.

La palabra más comúnmente utilizada en informática es el octeto que, en el caso de los ordenadores personales, llamamos **byte**. Un byte es simplemente una palabra de 8 bits, es decir, de 8 cifras binarias (cero o uno). Si quisiéramos representar los números decimales con un byte, ésta podría ser la tabla correspondiente:

Número decimal	Secuencia binaria
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
8	00001000
9	00001001

Vamos al fin a ver algo directamente relacionado con TCP. Volvamos al último artículo de la revista, y repasemos la cabecera TCP. Por si no lo tenéis a mano, aquí os la vuelvo a mostrar.

0 15 16 31
Puerto Origen | Puerto Destino
Numero de Secuencia
Numero de confirmacion
0 3 4 9 10 11 12 13 14 15
Comienzo de datos | 0 | URG | ACK | PSH | RST | SYN | FIN | Ventana
Suma de Comprobacion | Puntero de Urgencia
DATOS

Como vemos, en la cabecera TCP se manejan distintos tamaños de palabra. En primer lugar, para los campos **puerto origen** y **puerto destino** contamos con 16 bits, después, para los **números de secuencia y de confirmación** tenemos 32 bits, 4 bits para el campo **comienzo de datos**, 1 bit para cada uno de los **flags**, etc.

Para saber cuántos valores se pueden representar con cada tamaño de palabra, basta con hacer la **potencia de 2 con el tamaño de la palabra en bits**. Es decir, con una

palabra de 4 bits podemos representar 2^4 = 16 valores diferentes, con una palabra de 8 bits 2^8 = 256 valores, con 16 bits 2^{16} = 65536, etc.

Ahora podéis comprender por qué todo lo relacionado con la tecnología digital sigue siempre estos números. La memoria RAM que compras para el PC, las tarjetas de memoria para la cámara digital, la velocidad del ADSL, siempre son los mismos números; 8, 16, 32, 64, 128, 256, 512, 1024, 2048... Todos esos números son las diferentes potencias de 2.

El caso más sencillo es el de 1 único bit, donde tenemos $2^1 = 2$, es decir, se pueden representar sólo 2 valores con 1 bit. Estos 2 valores son, por supuesto: cero, y uno.

En el caso, por ejemplo, de 3 bits, tenemos 2^3 = 8 valores diferentes, que son todas las posibles combinaciones de 3 cifras, donde cada cifra puede ser un uno o un cero, tal y como vemos en la tabla.

Número decimal	Secuencia binaria
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Como vemos, no quedan más posibles combinaciones de ceros y unos con sólo 3 cifras, y esto nos permite representar tan sólo los números del 0 al 7.

Como ya dije antes, para un matemático estos convenios para representar los números decimales mediante cifras binarias no son ningún misterio, ya que basta con aplicar las bases de la aritmética modular.

Voy a tratar de explicar rápidamente en qué consisten estas fórmulas porque, aunque al principio os puedan parecer complicadas, en realidad son realmente sencillas y, como todo, es sólo cuestión de práctica el aplicarlas de forma natural.

2.1. Pasando de binario a decimal

Vamos a ver en primer lugar cómo traducir una secuencia de ceros y unos en algo comprensible para nuestras mentes decimales.

En la base decimal, que es la que nosotros utilizamos, llamamos a cada cifra de una manera diferente según el orden que ocupa: unidades, decenas, centenas, etc. Como nos explicaron en los primeros años del cole, para calcular un número a partir de su representación decimal, tenemos que sumar las unidades a las decenas multiplicadas por diez, las centenas multiplicadas por 100, etc., etc. Es decir: 534 = 5 * 100 + 3 * 10 + 4 * 1.

En realidad, 100 es una potencia de 10 (10^2 = 100). Y por supuesto 10 también es una potencia de 10 (10^1 = 10). pero también el 1 lo es, ya que 1 es potencia de cualquier número, pues $X^0 = 1$, donde en este caso, es X = 10, es decir, 10 elevado a cero es uno.

Por tanto, el número 534 se puede representar como: $534 = 5*10^2 + 3*10^1 + 4*10^0$.

Esta regla se puede aplicar a cualquier otra base que no sea 10. Volvamos a la tabla anterior, y veremos que el número 7 se representa como 111 en base 2.

Si aplicamos la fórmula anterior, pero en este caso utilizando base 2, tendremos: $1 * 2^2 + 1*2^1 + 1*2^0 = 7$. En efecto, se cumple, ya que $2^2 = 4$, $2^1 = 2$, y $2^0 = 1$, luego: 1*4 + 1*2 + 1*1 = 7.

Con esta sencilla fórmula de las potencias de 2 se puede convertir cualquier número binario a su equivalente en decimal. Por ejemplo, vamos a traducir a decimal el número 10011010.

Empezamos aplicando la fórmula: $1*2^7 + 0*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 0*2^0$. Ahora, sabiendo los valores de cada potencia de dos (cualquier *geek* que se precie tiene que conocer como mínimo todas las

potencias de 2 con exponentes de 0 a 16), podemos traducir esa fórmula en: 1*128 + 0*64 + 0*32 + 1*16 + 1*8 + 0*4 + 1*2 + 0*1. Es decir, nos queda la siguiente suma: 128 + 16 + 8 + 2 = 154. Por tanto, el número binario 10011010 representa al número 154 en decimal.

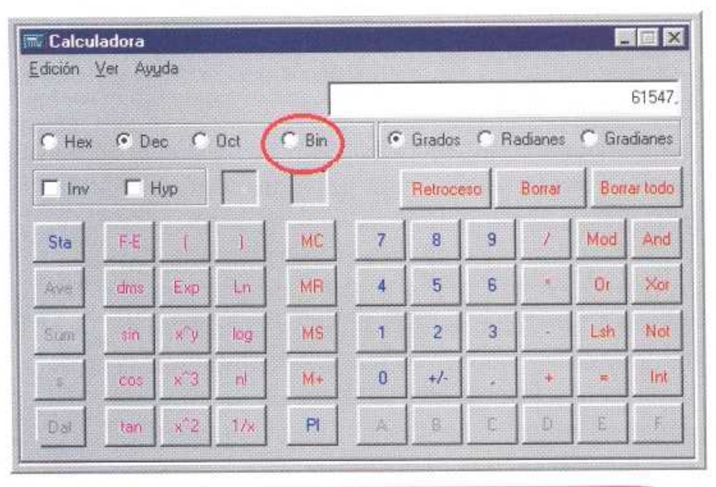
Por si queréis practicar, os dejo como ejercicio algunos números más, con su traducción, para que lo comprobéis vosotros mismos:

10110101 = 181
00111100 = 60
111010001010101010100101 = 15248037

2.2. Pasando de decimal a binario.

Aquí la cosa ya se pone más chunga. Aun así, juraría que esto ya lo expliqué en alguno de mis artículos.

Hay varios trucos para convertir de decimal a binario. El más sencillo, por supuesto, es meter el número en la calculadora de windows, y luego pinchar en donde pone BIN para que lo pase automáticamente, jeje.





Para que...

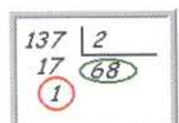
Para que a nadie se le ocurra enviar un mail diciendo que la calculadora de Windows no puede hacer eso, venga, lo explicamos muy rápido. Abre la calculadora, Menu Ver y pulsa sobre Científica. Ya está

Ahora introduce cualquier número y pulsa sobre Bin

Pero nosotros no nos conformamos con hacer las cosas, si no que nuestro auténtico interés es el saber cómo se hacen. Así que os explico rápidamente un algoritmo para convertir cualquier número decimal a binario.

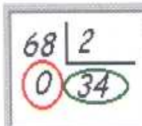
Usemos para el ejemplo el número 137.

El proceso a seguir será ir dividiendo **el número** (en este caso 137) **por 2** y, en cada división, quedarnos con el **resto** de la división (que sólo podrá ser cero o uno). Ahora te quedará claro.



El resultado de la primera división (llamado cociente, en verde) es 68, y el resto (en rojo) es 1. Este 1 será el bit menos significativo, es decir, la cifra binaria que está a la derecha del todo.

Continuamos el proceso con el nuevo cociente que hemos obtenido:



Ahora hemos obtenido un 0, que será la siguiente cifra binaria. Continuamos el proceso con el nuevo cociente, 34:

34 / 2 = 17, con resto 0.
17 / 2 = 8, con resto 1.
8 / 2 = 4, con resto 0.
4 / 2 = 2, con resto 0.
2 / 2 = 1, con resto 0.

Esta última operación (en azul) es fundamental, ya que el bit más significativo, es decir, el que hay más a la izquierda, será el último cociente, es decir 2/2 = 1.

Por tanto, el número 137 en binario nos quedará: **1 0 0 0 1 0 0 1**.



Si tienes...

Si tienes interés en avanzar por tu cuanta en cálculo binario, hay miles de páginas en Internet que te lo explican perfectamente y por supuesto de forma gratuita. Busca en www.google.com y avanza tanto como quieras

Construyendo un paquete TCP desde cero.

Ya podemos ponernos manos a la obra con el tema que nos ocupa, que son los paquetes TCP. Vamos a recordar la cabecera TCP (volved atrás un poco para ver la imagen), y a ir campo por campo construyendo el paquete.

Vamos a poner como ejemplo, el primer paquete que se envía cuando queremos establecer una conexión con un servidor de **FTP**.

En primer lugar, tenemos que conocer los **puertos de origen y de destino** (los dos primeros campos de la cabecera TCP). El puerto de **destino** será el **21**, que es el asignado por el estándar al servicio de **FTP** (aunque bien sabréis muchos de vosotros que no siempre se usa este puerto, como en el caso de los dumps, donde se suelen usar otros puertos menos "sospechosos").

El **puerto de origen** será un puerto aleatorio asignado por el sistema operativo a la hora de abrir el socket, es decir, la estructura utilizada por el sistema para establecer la nueva conexión TCP/IP. Supongamos que el sistema nos ha asignado el puerto **1345** como puerto de origen.

Ya tenemos los datos necesarios para rellenar la primera fila de la cabecera TCP. En primer lugar, tenemos que convertir el número 1345 en su equivalente binario, y lo mismo con el número 21.

Aplicando el algoritmo explicado en el punto anterior, obtenemos:

1345 = 10101000001
21 = 10101

Para construir la primera fila de la cabecera TCP tenemos que concatenar el puerto origen con el puerto destino, por lo que quedaría: 10101000001 10101.

Tenemos, por tanto, que nuestra primera fila consta de 16 bits... pero... no puede ser, si habíamos quedado en que cada fila de la cabecera TCP eran 32 bits.

El problema es que no hemos ajustado cada campo a su **tamaño de palabra**. El tamaño de palabra de cada uno de estos dos campos es de **16 bits**, por lo que cada uno de los dos números obtenidos tiene que ser representado con 16 cifras binarias. Para ello, habrá que poner el suficiente número de **ceros a la izquierda**, para rellenar las 16 cifras. Así, nos quedará:

1345 = 0000010101000001
21 = 0000000000010101

Ahora ya si que podemos concatenar ambos para conseguir la **primera fila** de la cabecera:

00000101010000010000000000010101

Como experimento, probad a convertir este número en su equivalente decimal. El resultado es 88145941. ¿Y qué nos dice este número? Pues absolutamente nada, ya que lo importante a la hora de traducir un número de una base a otra no es sólo la secuencia de cifras, si no también el cómo se agrupen estas. Si agrupamos esta secuencia en grupos de 16, entonces si que tendrá un sentido, pero si la agrupamos en una única secuencia de 32 bits, el número resultante no tiene ningún interés para nosotros. Por tanto, **es absolutamente imprescindible conocer el tamaño de las palabras**.

Vamos ahora con la segunda fila de la cabecera. Como vemos, ahora nos toca el campo **número de secuencia**. Este número también será asignado por el sistema operativo (recordemos del artículo anterior que no conviene que sea 0 cada vez que se establece una nueva conexión). En nuestro ejemplo el sistema nos asignará el número 21423994. Lo convertimos a binario, y nos da el número: 1010001101110011101111010.

Este número es de 25 bits, por lo que habrá que poner 7 ceros a su izquierda, para completar los 32 bits de la palabra que corresponde a este campo. Por tanto, la **segunda fila** de nuestra cabecera TCP será:

00000001010001101110011101111010

En este caso, el número completo de 32 cifras sí que tiene significado para nosotros, ya que el tamaño de la palabra es precisamente de 32 bits.

La **tercera fila** corresponde al campo **número de confirmación**. En nuestro caso tiene que ser cero, ya que es una conexión que aún no se ha establecido, por lo que el primer paquete no llevará confirmación de otro paquete anterior. Para representar el 0 con 32 bits, basta con meter 32 ceros.

00000000000000000000000000000000

Ahora nos toca el campo **comienzo de datos**. Como ya vimos, el valor más habitual para este campo es 5, en el caso de que no haya ninguna opción. Pero nosotros vamos a incluir una opción, que ocupará 32 bits, como veremos más adelante. Como el campo Comienzo de datos indica el número de palabras de 32 bits que ocupa la cabecera TCP, al tener una palabra más para la opción, tendrá que ser 6, es decir: 110. Como el campo tiene una palabra de 4 bits, añadimos un cero a la izquierda: **0110**.

A continuación, la cabecera TCP tiene un campo vacío de 6 bits, que hay que rellenar con ceros: **000000**. Por tanto, de momento esta fila de la cabecera nos va quedando: **0110000000**.

Ahora le toca el turno a los **flags**. Como vimos en el artículo anterior, siempre que se desee establecer una nueva conexión el paquete ha de tener activado su flag **SYN.** El resto de flags estarán desactivados. Es decir, éste será el valor que tomarán todos los flags:

URG = 0
ACK = 0
PSH = 0
RST = 0
SYN = 1
FIN = 0

Si los colocamos todos juntitos en su orden nos quedará: **000010**.

Esto habrá que concatenarlo a lo que llevábamos ya construido de esta fila, por lo que nos quedaría: **0110000000000010**

Vamos ahora con el campo **tamaño de la ventana**. Un valor típico es, por ejemplo, 8192. Este número es una potencia de 2, concretamente 2^{13}. Por tanto, la traducción a binario es instantánea. Basta con poner 13 ceros a la derecha, y poner un único 1 a la izquierda del todo: 10000000000000. Esto nos da un número de 14 cifras, por lo que tenemos que ajustarlo al tamaño de la palabra de 16 bits con dos ceros a la izquierda: **0010000000000000**.

Por tanto, finalmente, la **cuarta fila** de la cabecera TCP nos quedará:

01100000000000100010000000000000

El próximo campo es el campo **suma de comprobación**, y es el que más quebraderos de cabeza nos va a dar. Si habéis seguido el resto del curso, habréis visto que hasta ahora he "eludido" un poco el tema, dándoos sólo una URL donde teníais un código en C ya hecho para calcular automáticamente los **checksums** (sumas de comprobación). Si lo hice así hasta ahora era porque sabía que más adelante llegaría el momento de enfrentarse cara a cara con los checksums, **y ese momento ya ha llegado.**

Quizá os estaréis preguntando, ¿y por qué hay que enfrentarse al checksum si tenemos ya un código que nos lo calcula? ¿Para qué sirven todas estas vueltas y revueltas que estoy dando a los paquetes TCP cuando bastaría con conocer lo necesario para poder manejarlos?

Creo que es importante que hablemos aquí acerca del significado original de la palabra ***HACK***, que forma parte del nombre de esta revista, y que justifica el hecho de que profundicemos hasta el más mínimo detalle en lo que explicamos.

El término *Hacker* ha sido muy desvirtuado con el paso del tiempo. Originalmente, un hacker era una persona cuya pasión era conocer el funcionamiento de las cosas. Para las personas "normales" una máquina es sólo una herramienta que se utiliza para algún fin concreto. En cambio, un hacker no se conforma sólo con usar las máquinas, si no que además ansía conocer su funcionamiento interno.

Hace años, se llamaba hackers a los grandes programadores, a los gurús de cualquier campo de la informática, y a toda esa clase de chiflados (a los cuales aspiro orgullosamente a pertenecer). Posteriormente, los medios de comunicación tergiversaron todo, y dieron a la palabra hacker el significado que antiguamente tenía la palabra cracker, y después estos términos han seguido evolucionando hasta el punto actual, en el cual hay mil y una definiciones para cada uno de los términos.

La cuestión es que si realmente queréis ser *hackers*, de los de toda la vida, vuestra pasión debe ser conocer hasta el mínimo detalle de cómo funcionan las cosas, y no sólo saber "manejarlas" sin más. Un tío que dedique a entrar en sitios donde teóricamente le estaba prohibido el paso, será un hacker sólo si su motivación para hacerlo sea explorar el funcionamiento de los sistemas, en caso contrario, su calificativo más apropiado será lamer, script kiddie, o el que más os guste.

Pero bueno, ya he vuelto a salirme del tema... estábamos con el checksum. Pues me temo que de momento tenemos que dejar este punto en blanco, porque para calcular el checksum tenemos que tener terminada el resto de la cabecera, así que vamos a ver antes el resto de campos, y luego volvemos atrás sobre este punto.

El siguiente campo es el **puntero de urgencia**. Como el flag **URG** no está activo, este campo puede ser 0. Como son 16 bits, tendremos aquí: **0000000000000000**.

Por último, tenemos el campo **DATOS**. Como el paquete es sólo para establecer una conexión, no habrá ningún dato, por lo que este campo estará en blanco (ya no con ceros, si no que directamente no habrá **nada**).

Pero... ¡un momento! ¡Si habíamos dicho que íbamos a meter una opción! Entonces el campo DATOS no será el último, si no que tendremos antes el campo de **opciones TCP**. Para el caso nos va a dar igual, porque al fin y al cabo no hay campo de DATOS, así que en cualquier caso el campo **opciones** iría inmediatamente después del campo puntero de urgencia.

La opción que vamos a incluir es la única definida en el RFC de TCP, aunque ya vimos que existen muchas más: **Maximum Segment Size (MSS).**

Todas las opciones empiezan con un byte que indica el **tipo de opción**. En nuestro caso, el código para la opción MSS es el **2,** es decir: **00000010**.

En el caso de la opción MSS, el siguiente byte contendrá la **longitud en bytes de la opción** que, contando con los dos primeros bytes que ya hemos mencionado (el que indica el código, y el que indica la longitud) será siempre **4.** Por tanto, el segundo byte de la opción MSS será siempre fijo: **00000100**.

Por último, los otros dos bytes que completarían la fila de 32 bits serán los que contengan el dato que queremos transmitir: el **tamaño máximo de segmento**. Si, por ejemplo, queremos un tamaño máximo de segmento de 1460 bytes, codificaremos este valor en 16 bits: **0000010110110100**.

Por tanto, toda la **fila de 32 bits para la opción MSS** nos quedaría: **00000010000001000000010110110100**.

Calculando la suma de comprobación (checksum).

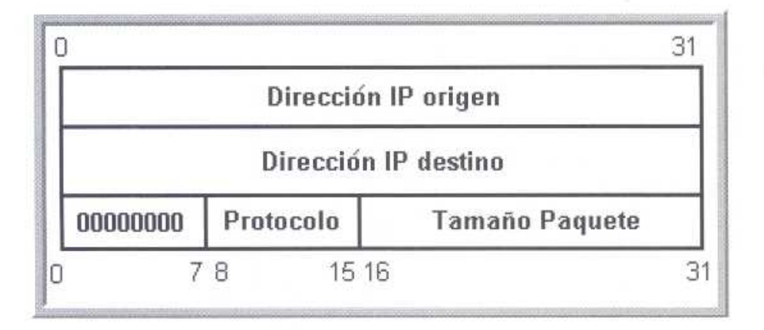
Ya podemos volver atrás un poco y calcular el último campo que nos falta para completar

la cabecera TCP de nuestro paquete. El primer paso a seguir es coger todo lo que tenemos hasta ahora, y agruparlo en palabras de 16 bits. Es decir, partimos de todos estos chorizos binarios:

0000 0101 0100 0001 = puerto de origen
0000 0000 0001 0101 = puerto de destino
0000 0001 0100 0110 = primeros 16 bits del número de secuencia
1110 0111 0111 1010 = últimos 16 bits del número de secuencia
0000 0000 0000 0000 = primeros 16 bits del número de confirmación
0000 0000 0000 0000 = últimos 16 bits del número de confirmación
0110 0000 0000 0010 = comienzo de datos, y flags
0010 0000 0000 0000 = tamaño de la ventana
0000 0000 0000 0000 = puntero de urgencia
0000 0010 0000 0100 = código y longitud de la opción MSS
0000 0101 1011 0100 = opción MSS

Por si todas estas ristras de ceros y unos os parecen pocas, todavía tenemos que añadir unas cuantas más, y es aquí cuando entra en juego esa pequeña cabecera de la que os hablé que se utilizaba a la hora de calcular el checksum.

Recordemos esta cabecera:



En primer lugar, necesitamos la **IP de origen**. Supongamos que tenemos un router ADSL que nos conecta con Internet, por lo que nuestra IP será una IP de red local, como por ejemplo: **192.168.1.1**.

Si, por ejemplo, el FTP al que conectamos es el de **Rediris** (ftp.rediris.es) sabemos que su **IP** es **130.206.1.5**.

El número de **protocolo** asignado a **TCP** es el **6.**

Por último, el tamaño del paquete TCP lo calculamos contando todos los bytes que hay en la **cabecera TCP**, y todos los bytes de **DATOS**. Como en nuestro paquete no hay datos, bastará con contar los bytes (grupos de 8 bits) que ocupa la cabecera. Cada fila de la cabecera son 4 bytes (32 / 8 = 4), y tenemos un total de 6 filas, por lo que el tamaño de paquete será de 6 * 4 = **24**.

Ahora tenemos que pasar todo esto a binario:

IP de origen = 1100 0000 . 1010 1000 . 0000 0001 . 0000 0001
IP de destino = 1000 0010 . 1100 1110 . 0000 0001 . 0000 0101
Protocolo = 0000000000000110
Tamaño de paquete = 0000000000011000

Lo que hay que hacer ahora con todos estos chorizos binarios es simplemente **sumarlos** (de ahí el nombre de "suma" de comprobación). El problema es que, si no tenéis práctica, sumar en binario os puede resultar complicado. Sería ya demasiado explicaros ahora toda la aritmética binaria, así que eso os lo dejo como ejercicio para que lo estudiéis por vuestra cuenta (www.google.com o utiliza la calculadora de Windows).

Lo que voy a hacer yo es pasar todo esto a hexadecimal para manejar menos cifras engorrosas. Lo que me queda al final es todo esto:

C0A8 + 0101 + 82CE + 0105 + 0006 + 0018 = 1459A

Este primer resultado es la suma de toda la pseudocabecera de checksum que acabamos de calcular. Ahora hay que hacer otra suma, pero con todos los chorizos que sacamos antes, de la propia cabecera TCP:

0541 + 0015 + 0146 + E77A + 0000 + 0000 + 6002 + 2000 + 0000 + 0204 + 05B4 = 175D0.

Ahora sólo tenemos dos números, que tendremos que sumar a su vez:
1459A + 175D0 = 2BB6A

Este no es todavía el resultado, entre otros motivos porque el checksum ha de ocupar

sólo **16 bits**, y un número hexadecimal de 5 cifras, como el 2BB6A, ocupa 20 bits. Por tanto, lo que hacemos es coger la primera cifra (el 2) y sumarla al resto:

BB6A + 2 = BB6C

Ya tenemos completada la operación conocida como **suma en complemento a uno**.

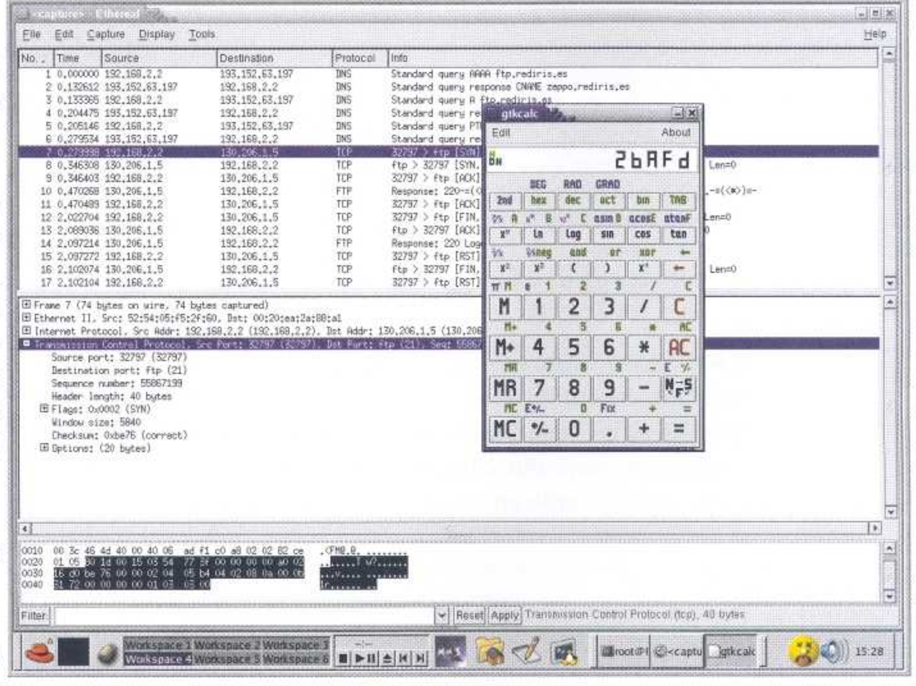
Ahora sólo nos falta sacar a su vez el **complemento a uno** de este número, es decir, **invertir todos los bits** (donde haya un uno, poner un cero, y viceversa). Si pasamos este número (BB6C) a binario tenemos: **1011 1011 0110 1100**.

Si invertimos cada bit nos queda:
0100 0100 1001 0011
Este número de **16 bits** es al fin lo que tenemos que poner en el campo que nos quedaba por rellenar: la **suma de comprobación**.

Repasamos por tanto todo lo visto acerca de la suma de comprobación:

1- **Agrupamos** toda la cabecera TCP en grupos de **16 bits**, y **sumamos** todos esos grupos entre sí.
2 - A continuación, **construimos la pseudocabecera** con las ips de origen y de destino, el protocolo, y el tamaño de paquete, y **agrupamos** todo esto también en grupos **de 16 bits**.
3 - **Sumamos** estos nuevos grupos al resultado obtenido en el primer paso.
4- Del número obtenido en el paso 3, nos quedamos sólo con los **16 últimos bits**, y **los sobrantes los sumamos** a estos 16, quedándonos así un resultado de **16 bits**.
5- El resultado del cuarto paso, lo **invertimos**, cambiando cada cero por un uno, y viceversa.

¿Que cómo he hecho todas esas sumas en hexadecimal? Pues, por supuesto, no de cabeza, si no usando una calculadora científica que admita hexadecimal.



En esta captura podemos verme en plena faena, calculando el checksum de un paquete capturado con un sniffer. El sniffer me da directamente el paquete en hexadecimal, por lo que me facilita los cálculos.

¿Y cuál es la utilidad de todas estas operaciones?
Pues para comprenderlo necesitáis saber que el complemento a uno de un número es el **opuesto** de ese número en complemento a uno, es decir, si sumas en complemento a uno un número cualquiera y su complemento a uno, el resultado tiene que ser siempre **cero**. Vamos, en resumen, que es como decir que -5 es el complemento a uno de 5, ya que 5 + (-5) = 0.

Esto hace que el procesamiento de los paquetes sea bastante rápido a la hora de verificar los checksum, ya que basta con que el software sume todos los datos del paquete y, si el paquete es correcto, el resultado tiene que ser 0.

Esto es lógico, ya que la suma que nosotros hemos hecho hace un momento contenía todos los datos del paquete excepto uno: la propia suma de comprobación. El paquete que llegue al receptor, en cambio, si que contendrá además ese dato, y como ese dato es precisamente el opuesto de la suma de todos los demás, al sumar todos los datos más la suma de comprobación, el resultado será:

checksum + resto de cabecera = 0

Ya que, insisto:

checksum = - (resto de cabecera).

Os propongo como ejercicio que comprobéis todo esto con un sniffer. Capturad un paquete TCP, sumad todos los datos de la cabecera TCP, del campo DATOS, y de la pseudocabecera utilizada en el checksum, y comprobaréis que, si el paquete no contiene errores, el resultado es siempre cero.

Resumiendo

Al final, este es el paquete que nos queda, y que será enviado tal cual desde nuestro PC hasta el receptor (en este caso, el servidor FTP de Rediris):

0000 0101 0100 0001								0000 0000 0001 0101
0000 0001 0100 0110 1110 0111 0111 1010								
0000 0000 0000 0000 0000 0000 0000 0000								
0110 000000		0	0	0	0	1	0	0010 0000 0000 0000
0100 0100 1001 0011								0000 0000 0000 0000
0000 0010	0000 0100							0000 0101 1011 0100

4. La respuesta a nuestro paquete

Al recibir este paquete el servidor FTP de Rediris, nos responderá con el siguiente paquete, que os muestro como ejercicio para que lo analicemos:

0000 0000 0001 0101								0000 0101 0100 0001
0010 0111 1001 0010 1000 0000 0101 0011								
0000 0001 0100 0110 1110 0111 0111 1011								
0110 000000		0	1	0	0	1	0	0011 0000 0010 0110
0111 1101 0000 0010								0000 0000 0000 0000
0000 0010	0000 0100							0000 0101 1011 0100

Vamos a analizarlo:

En primer lugar, los **puertos de destino y de origen** están **intercambiados**, como es lógico.

En segundo lugar, vemos que el **número de secuencia** no tiene nada que ver con el nuestro, ya que cada extremo de la comunicación usará sus propios números de secuencia.

En cambio, el que sí que tiene que ver con nuestro número de secuencia es **su número de confirmación**. Al no contener datos nuestro paquete, el próximo byte que enviaríamos sería el inmediatamente posterior al número de secuencia de nuestro paquete anterior. Por tanto, el servidor de Rediris estará esperando recibir en el próximo paquete un número de secuencia que sea el que enviamos antes, + 1.

Vemos que su campo **comienzo de datos** es el mismo que el nuestro, ya que el paquete también contendrá una única fila de opciones (de 32 bits).

Donde vemos que sí que hay un cambio es en los **flags,** ya que aquí no sólo está activado el flag **SYN**, si no también el flag **ACK**. En el próximo punto veremos en detalle a qué se debe esto.

Vemos también que el **tamaño de la ventana** es diferente, ya que cada extremo de la comunicación puede tener su propio tamaño de ventana.

La **suma de comprobación**, por supuesto, es diferente para cada paquete. Os propongo como ejercicio que comprobéis la validez de la suma de comprobación de este paquete y, en caso de que sea incorrecta, que calculéis cuál sería el checksum correcto.

El **puntero de urgencia** también está a cero, ya que tampoco tiene el flag **URG** activado.

Por último, vemos que también añade la **opción MSS**, con el mismo tamaño máximo

de segmento: 1460 bytes.

El paquete tampoco contiene nada en el campo **DATOS**, ya que es otro de los paquetes utilizado únicamente para establecer la conexión.

5. Los estados de conexión TCP

No podemos completar un curso sobre TCP sin hablar de los estados de conexión. Para ello, empezaremos viendo una serie de procedimientos que se llevan a cabo en las conexiones TCP, para luego enumerar los estados que hemos ido descubriendo.

5.1. Establecimiento de conexión TCP.

En el ejemplo anterior, los paquetes involucrados correspondían a un establecimiento de conexión. Como vimos, había por lo menos dos paquetes encargados de establecer la conexión: uno por nuestra parte, que le decía al servidor de Rediris que queríamos establecer la conexión, y otro por parte del servidor de Rediris que nos decía que estaba de acuerdo y él también quería establecer la conexión.

En realidad, el establecimiento de conexión TCP requiere aún otro paquete más, y es lo que hace que a este sistema de establecimiento de conexión se le llame **3-way handshake** o, traducido así a lo bruto: saludo de 3 pasos.

El sistema de establecimiento de conexión TCP consiste en lo siguiente:

1- El cliente solicita al servidor una conexión.

2- El servidor responde aceptando la conexión.

3- El cliente responde aceptando la conexión.

¿A qué se debe la necesidad de este tercer paso? Este tercer paso permite una sincronización exacta entre cliente y servidor, y además permite al cliente "echarse atrás" si no le gusta algo en la respuesta del servidor.

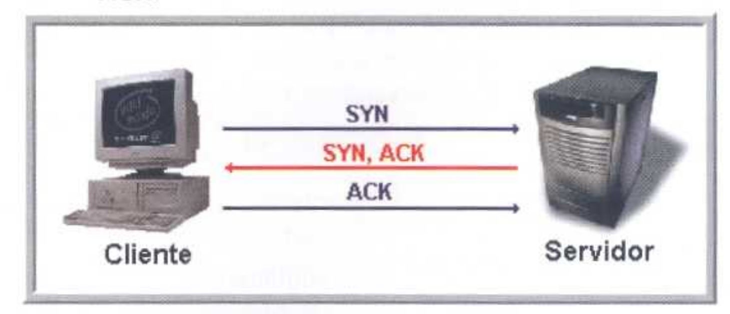
Por ejemplo, en el caso del servidor de Rediris, si habéis probado a conectaros habréis visto que nada más conectar envía un texto de presentación. Si nosotros no respondiésemos al servidor diciéndole que aceptamos la conexión, aún habiéndola solicitado nosotros, el servidor se pondría en vano a enviar todo ese texto sin saber si al otro lado hay alguien escuchando.

Estos tres paquetes especiales utilizados en el 3-way handshake se caracterizan por sus flags:

El cliente solicita conexión al servidor: flag **SYN**.

El servidor acepta la conexión: flags **SYN** y **ACK**.

El cliente acepta la conexión: flag **ACK**.



Esto ha de ser siempre así, y si alguno de esos flags no es enviado en el orden correcto, todo el establecimiento de conexión será anulado.

Intuitivamente, nos damos cuenta de que todo este mecanismo nos da lugar a diferentes estados en la conexión:

En primer lugar, el estado primordial es el estado **DESCONECTADO**, que es cuando aún ni siquiera hemos enviado el SYN.

En segundo lugar, una vez enviado el SYN, estamos en un estado diferente que espera al próximo paso. A este paso lo podemos llamar **SYN ENVIADO**.

Una vez que el servidor recibe el primer SYN, nos enviará su respuesta con el SYN y el ACK en el mismo

paquete. En ese caso, pasaremos a un estado que podemos llamar **SYN RECIBIDO**. Ahora nos falta esperar al último paso del establecimiento, que es el último ACK.

Una vez recibido el último ACK, nuestra conexión pasará finalmente al estado de conexión **ESTABLECIDA**.

DESCONECTADO
ESCUCHANDO
SYN ENVIADO
SYN
SYN, ACK
SYN RECIBIDO
ESTABLECIDA
ACK
Cliente
ESTABLECIDA
Servidor

Si alguna vez habéis utilizado la herramienta **netstat** posiblemente os suenen estos nombres. Si no, probad ahora mismo, desde una shell de Linux/Unix, o una ventana Ms-DOS de Windows, a escribir:

netstat

Veréis la lista de conexiones de vuestra máquina, con el estado de cada una. Las conexiones que estén en estado ESTABLECIDA, que son las más habituales, aparecerán como **ESTABLISHED**.

Podréis ver otros estados como **CLOSE_WAIT, FIN_WAIT, LISTEN**, etc. En breve explicaremos todos ellos.

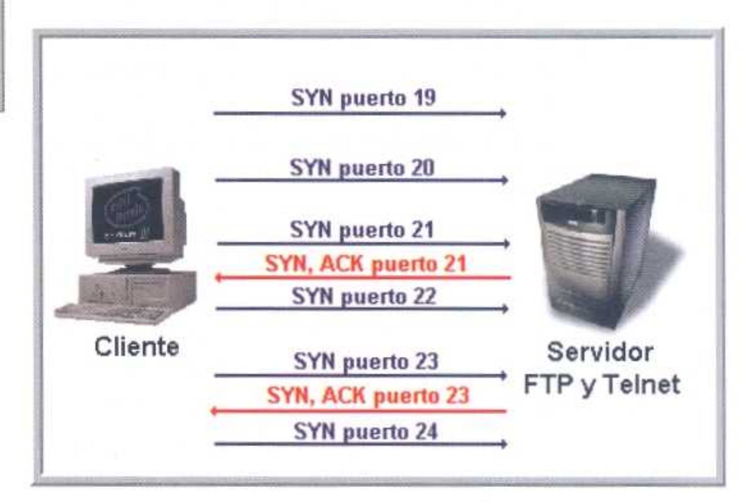
Escaneo de puertos con SYN

Os propongo como experimento que pongáis en marcha alguna aplicación de escaneo de puertos, y a continuación hagáis un **netstat**.

Probablemente (dependiendo del tipo de escaneo que utilice la aplicación), veréis que hay montones de conexiones en estado **SYN_SENT**, es decir, lo que nosotros hemos llamado SYN ENVIADO.

Esto se debe a que un sistema clásico de escaneo consiste en hacer solicitudes al servidor para establecer conexiones en cada uno de los puertos, es decir, enviamos un paquete **SYN** a cada puerto. Los paquetes que tengan como respuesta un **SYN, ACK** corresponderán a los puertos abiertos de la máquina. Los paquetes que no tengan respuesta, o bien que sean respondidos con un flag **RST**, estarán cerrados.

Lo interesante aquí es que nosotros no responderemos a ninguno de los **SYN, ACK**, por lo que ninguna conexión quedará establecida, ya que sería necesario que respondiésemos con un nuevo paquete **ACK** por cada paquete **SYN** que enviásemos.



Ataque SYN Flood

Ya que hemos empezado hablando de cosas divertidas, como el escaneo de puertos, vamos a rematar la faena hablando de una técnica de hacking realmente interesante, aunque más por el interés de su funcionamiento que por su utilidad práctica, ya que es un ataque de tipo **DoS** (**D**enial **o**f **S**ervice), es decir, que sólo sirve para fastidiar y tirar abajo un servidor.

¿Por qué hablamos de diferentes estados en una conexión? Pues porque es necesario tener en cuenta los diferentes estados a la hora de llevar a cabo todos los pasos necesarios para conseguir llevar a cabo la comunicación.

Por ejemplo, en el momento en que pasamos al estado **SYN_SENT**, tenemos que crear una **estructura de datos** que necesitaremos para mantener controlado el estado de la conexión en todo momento. Sería absurdo tener estos

datos creados de antemano, ya que ocuparían una gran cantidad de memoria innecesaria y, además, tampoco podríamos saber cuántas estructuras de este tipo necesitaríamos, pues depende en todo momento de cuántas conexiones tengamos establecidas. Por tanto, al cambiar un sistema al estado SYN_SENT, creará una estructura de datos para mantener la conexión inminente. Esta estructura de datos se llama **TCB** (**T**ransmision **C**ontrol **B**lock).

Por tanto, cada vez que intentamos conectar con un servidor, estamos haciéndole crear una estructura que ocupa lugar en su memoria. En el momento en que se cierre esa conexión, el servidor podrá borrar el TCB correspondiente, recuperando así el espacio que había ocupado en su memoria.

¿Qué pasaría entonces si saturásemos al servidor a base de conexiones? Si esta saturación es suficiente, conseguiremos dejar sin memoria al servidor para establecer nuevas conexiones.

Para que esta saturación sea efectiva, es conveniente que utilicemos direcciones IP falsas, es decir, que hagamos un **IP Spoofing**.

Si conseguimos enviar al servidor una gran cantidad de paquetes, cada uno de ellos conteniendo un flag **SYN** solicitando una conexión, y cada uno con una dirección IP falsa, el servidor creará un TCB para cada una de esas conexiones falsas inminentes y, al mismo tiempo, enviará el **SYN, ACK** a cada una de las direcciones falsas.

A continuación, se quedará esperando a recibir el **ACK** para completar el establecimiento de conexión de cada una de las conexiones falsas.

Por supuesto, este **ACK** jamás llegará, y el servidor se quedará esperando hasta que se canse. Lo malo es que los servidores son bastante pacientes, y suelen esperar en torno a unos 3 minutos antes de dar por perdida una conexión. Por tanto, si saturamos al servidor a base de **SYN**s, en unos 3 minutos nadie podrá conectarse legítimamente al servidor, ya que su memoria para nuevas conexiones estará llena en espera de completar las que tiene pendientes.

Si este bombardeo de **SYN**s se repite constantemente, el servidor quedará inutilizado mientras dure el bombardeo.

Si leísteis mi artículo de la serie RAW sobre el protocolo **DNS**, o mi artículo sobre **UDP** del curso de TCP/IP, conoceréis la técnica de envenenamiento de caché DNS. Cuando expliqué esta técnica mencioné que una ayuda para hacer más efectivo el ataque era conseguir hacer una denegación de servicio (DoS) al servidor DNS legítimo.

En cambio, la técnica de **SYN Flood** no puede funcionar en UDP, ya que sencillamente UDP no tiene ningún flag, y menos aún el flag SYN, por lo que en este caso el utilizar UDP frente a TCP es una ventaja para el protocolo DNS.

Más adelante veremos cómo llevar a cabo un ataque SYN Flood en detalle de forma práctica.

5.2. Cierre de conexión TCP

Continuando con el asunto de los estados de conexión en TCP, vamos a ver otro procedimiento que se puede llevar a cabo con cualquier conexión, y es el cierre de la misma.

Si pensamos un poco, nos daremos cuenta de que hay un pequeño problema inherente a cualquier **conexión full-duplex**, es decir, las conexiones en las que cualquiera de las dos partes puede tanto transmitir como recibir. El problema es que, si ambos quieren transmitir datos, ambos tendrán que ponerse de acuerdo para decidir en qué momento hay que cerrar la conexión.

Si no hubiese esta clase de acuerdos, ocurrirían cosas poco deseables, como por ejemplo que conectásemos con un FTP, solicitásemos un archivo, y tras enviárnoslo el servidor nos cerrase la conexión, asumiendo que ya no queremos bajar ni subir nada más. O, por ejemplo, conectarnos a un chat, decir "*hola*", y que el servidor decidiese que ya no queremos decir nada más y, por tanto, nos cerrase la conexión.

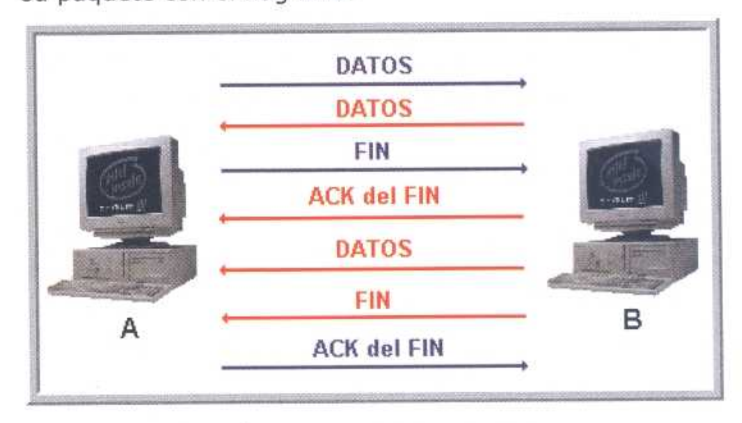
Por tanto, en una conexión full-duplex es necesario que ambas partes se pongan de acuerdo sobre el momento en el que hay que cerrar la conexión.

En el caso de TCP, el sistema que se utiliza para conseguir esto es sencillamente que cada uno, por su cuenta, indique al otro el momento en el que quiere cerrar la conexión.

Una vez que el otro se ha enterado, habrá que esperar a que él también desee cerrar la conexión. Es decir, si hemos terminado de enviar datos, decimos "*Por mi ya está todo hecho. Avisame cuando termines tú*". En el momento en que el otro termine, avisará diciendo: "*Vale, yo también he terminado, así que hasta luego*".

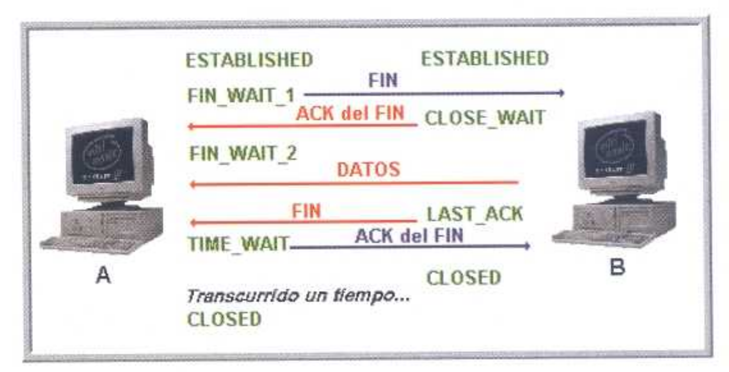
Para dar este tipo de avisos lo que se hace es enviar un paquete especial que tiene activado un flag que sirve precisamente para indicar que deseamos **FINalizar** la conexión. Este flag es el flag **FIN**.

A partir del momento en que enviamos un paquete con el flag **FIN**, ya **no debemos enviar** ningún paquete más (excepto en el caso de que tuviéramos que reenviar un paquete anterior porque no recibiésemos su confirmación), y lo único que debemos hacer es **seguir recibiendo** los datos de nuestro compañero, hasta que éste también nos envíe su paquete con el flag **FIN**.



Esto, una vez más, nos da lugar a nuevos estados de conexión. Una vez que enviamos nuestro **FIN** entramos en un estado en el que no podemos enviar, y sólo podemos recibir. A este estado lo podemos llamar **ESPERA DEL FIN**.

En realidad, el cierre de conexión da lugar a varios estados diferentes, que podemos ver en la siguiente imagen:



Aquí he puesto ya los nombres auténticos de los estados, en inglés, porque sería demasiado rebuscado tratar de traducirlos, jeje.

Como vemos, el estado que he llamado antes ESPERA DEL FIN es el que se llama **FIN_WAIT_1**.

En el momento en que nuestro compañero recibe nuestro **FIN,** él entra en otro estado diferente, que es el **CLOSE_WAIT**, es decir,

en espera de cerrar, ya que es ahora responsabilidad suya terminar de enviar cuando pueda, para enviar él también su **FIN**.

Por otra parte, cuando recibe nuestro **FIN**, tiene que confirmarnos su recepeción, igual que con cualquier otro paquete, enviándonos un **ACK**.

En el momento en que recibimos ese **ACK**, entramos en otro estado, que es el **FIN_WAIT_2**.

En el momento en que nuestro compañero termina, éste envía su **FIN**, y queda en espera de recibir nuestra confirmación. Durante esta espera, entra en estado **LAST_ACK**.

Una vez que recibimos ese **FIN**, entramos en estado **TIME_WAIT**, y enviamos el **ACK** para el **FIN** de nuestro compañero.

Una vez completado todo esto, ambas partes pasan a estado **CERRADO**, y se acabó el tema. La máquina **A** esperará un tiempo prudencial para pasar a estado **CERRADO**, ya que no puede estar seguro de que **B** haya recibido correctamente su último **ACK**.

5.3. Lista de estados TCP

Ya podemos ver la lista completa de estados de conexión en TCP, que os servirá como guía de referencia, sobre todo para cuando utilicéis herramientas como **netstat**.

LISTEN - Es el estado en el que permanece cualquier servidor cuando está en espera a que un cliente se conecte. Todos los puertos que tengamos abiertos en nuestro PC nos generarán un socket TCP (o UDP, según el caso) en estado LISTEN. Cada vez que un cliente se conecte, si permitimos más de una conexión, se creará un socket en estado ESTABLISHED para ese cliente, y otro en estado LISTEN para esperar al resto de clientes.

SYN_SENT - Se entra en este estado cuando solicitamos una conexión a un servidor (enviamos un paquete SYN), y aún no hemos recibido su aceptación o su rechazo. *(Ver establecimiento de conexión).*

SYN_RECEIVED - Se entra en este estado cuando tanto cliente como servidor han enviado sus correspondientes SYN, y estamos en espera de que se complete el tercer y último paso del establecimiento de conexión. *(Ver establecimiento de conexión).*

ESTABLISHED - Conexión establecida. Es el estado habitual.

FIN_WAIT_1 - Hemos enviado un paquete FIN, y sólo podemos recibir, pero no enviar. Estamos esperando a que nuestro compañero nos confirme que ha recibido nuestro FIN. Queremos cerrar la conexión, pero aún no sabemos si nuestro compañero se ha enterado. *(Ver cierre de conexión).*

FIN_WAIT_2 - Hemos enviado un paquete FIN, y sólo podemos recibir, pero no enviar, pero además hemos recibido ya la confirmación (ACK) de nuestro FIN. Por lo tanto, sabemos ya que nuestro compañero conoce nuestras intenciones de cerrar la conexión. *(Ver cierre de conexión).*

CLOSE_WAIT - Hemos recibido el FIN de nuestro compañero, pero a nosotros todavía nos quedan datos por enviar. *(Ver cierre de conexión).*

CLOSING - Esperamos a la última confirmación para cerrar definitivamente la conexión. Es un estado al que se llega cuando ambas partes desean cerrar la conexión simultáneamente, al contrario del caso que explicamos anteriormente.

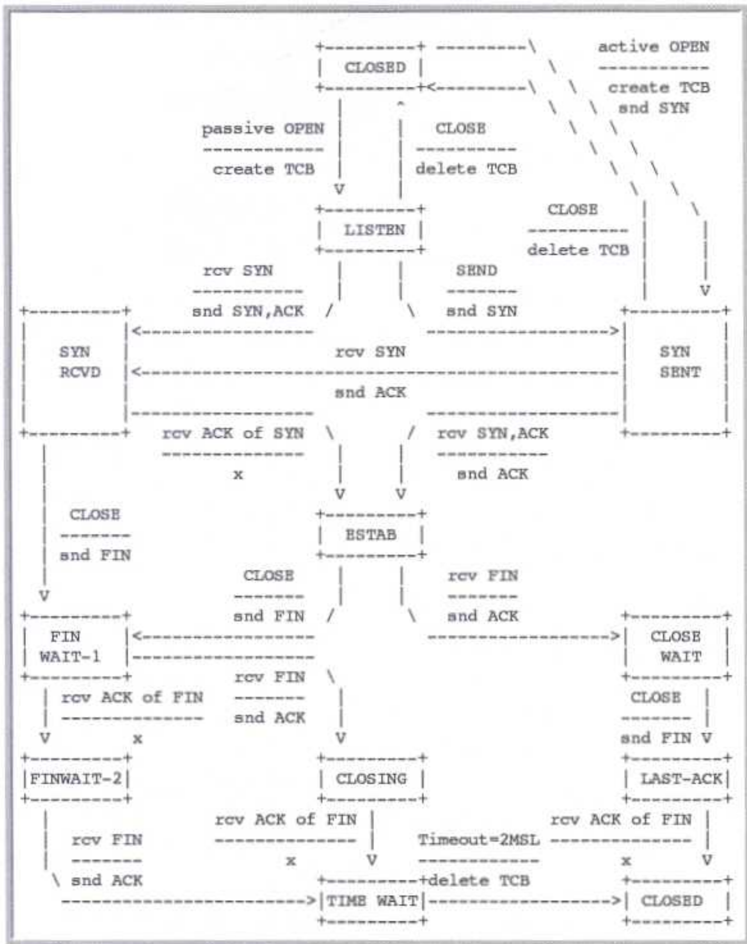
LAST_ACK - Esperamos a la confirmación (ACK) de nuestro FIN, cuando eramos nosotros los últimos que faltabamos por enviar el FIN. *(Ver cierre de conexión).*

TIME_WAIT - Hemos enviado la confirmación del FIN a nuestro compañero, cuando era él el que faltaba por enviar el FIN. Se llama así, porque lo que hacemos es esperar un tiempo prudencial para asumir que ha recibido nuestra confirmación. Siempre que nosotros cerremos

una conexión, durante un tiempo permaneceremos en estado TIME_WAIT, por lo que es bastante común encontrar este estado cuando hacemos netstat. *(Ver cierre de conexión).*

CLOSED - Es un estado ficticio, que simplemente dice que no hay ningún tipo de conexión.

A continuación, muestro el diagrama de estados de TCP, tal y como lo podéis encontrar en el RFC:



6. RAW Sockets TCP (Nemesis y Hping)

Vamos a recordar un poco las herramientas que expliqué para manejar RAW sockets en el artículo sobre UDP, pero en este caso aplicadas al caso de TCP. Esta vez empezaremos por Linux.

6.1. Hping2 para Linux

El caso de TCP es bastante más complicado que el de UDP, ya que TCP da muchísimo más juego. No hay más que ver el manual:

man hping2

Para ver la gran cantidad de opciones que hay para TCP que, por cierto, es el protocolo por defecto de Hping2. Nosotros vamos a ver sólo las opciones que directamente tienen que ver con lo explicado hasta ahora.

Empezamos con la prueba más sencilla: **hping2 130.206.1.5 --destport 21 --count 1**

Con esto enviamos un único (--count 1) paquete TCP a la dirección 130.206.1.5, al puerto 21.

Este paquete tendrá todos sus parámetros tal y como los tiene configurados hping2 por defecto, es decir, sin ningún flag, con un tamaño de ventana de 64 bytes, y sin opciones.

Este paquete, por tanto, sirve para bien poco, aunque en el manual de hping2 nos explican que, si ni siquiera utilizamos el parámetro --destport, por defecto envía el paquete al puerto 0, y esto puede ser útil para hacer un "ping" a una máquina que tenga un firewall que filtre los auténticos pings (que no son TCP, si no ICMP, que es otro protocolo), y además es probable que éste intento nuestro de ping ni siquiera quede reflejado en los logs de la máquina. Aún así, esto es algo que yo no he comprobado, así que no sé qué utilidad tendrá. Os propongo que lo probéis vosotros mismos como ejercicio.

Vamos a ver las diversas opciones que nos da Hping2 para TCP:

--baseport: permite especificar nuestro puerto de origen.
--destport: permite especificar el puerto de destino.
--keep: si enviamos varios paquetes, evita

que el puerto de origen se vaya incrementando automáticamente, tal y como vimos con UDP.
--win: fija el tamaño de la ventana TCP.
--tcpoff: envía un valor falso para el campo Comienzo de Datos de la cabecera TCP.
--tcpseq: especifica el Número de Secuencia.
--tcpack: especifica el Número de Confirmación.
--badcksum: igual que en UDP, envía un checksum erróneo.
--fin: el paquete que enviamos tiene activo el flag FIN.
--syn: el paquete que enviamos tiene activo el flag SYN.
--rst: el paquete que enviamos tiene activo el flag RST.
--push: el paquete que enviamos tiene activo el flag PUSH.
--ack: el paquete que enviamos tiene activo el flag ACK.
--data: especifica el tamaño del campo DATOS, sin contar con la cabecera.
--file: igual que en UDP, permite rellenar el campo DATOS con los contenidos de un archivo que especifiquemos.
--safe: nos permite asegurarnos de que los paquetes que enviamos llegan a su destino ya que, tal y como ha de hacerse en TCP, si no recibimos la confirmación de alguno de los paquetes, hping2 lo reenviará automáticamente.

Vamos a ver todo esto y mucho más con un ejemplo muy interesante, que es para poner en práctica la técnica de SYN Flood explicada anteriormente.

SYN Flood mediante Hping2.

Esta técnica sólo debéis utilizarla para hacer pruebas con vosotros mismos, para comprender el funcionamiento de la técnica, y también para poner a prueba la seguridad de vuestra red, por si queréis hacer una auditoría de seguridad y arreglar los agujeros que tengáis.

Cualquier otro uso que le deis, al margen de que pueda ser ilegal, éticamente será indeseable, ya que no estaréis más que fastidiando por fastidiar. Además, es poco probable que funcione un IP spoofig a pelo como el que voy a explicar, ya que los routers que haya en el camino desde vosotros hasta vuestra "víctima" probablemente rechacen los paquetes si no provienen de una IP que forme parte de su red.

Recordemos que para explotar la técnica de SYN Flood "simplemente" hay que enviar gran cantidad de paquetes con flag **SYN**, cada uno con una **dirección IP de origen falsa** y, a ser posible, diferente. Hping2 "casualmente" tiene opciones para automatizar todo esto, por lo que nos basta con esta línea:

hping2 192.168.1.1 --rand-source --destport 21 --syn --count 100

Con esta línea enviaremos 100 paquetes (--count 100) al puerto 21 de la IP 192.168.1.1, utilizando como IP de origen una aleatoria en cada paquete (--rand-source), y con el flag SYN activado.

Os puedo asegurar que esta línea funciona, ya que acabo de probarla ahora mismo con el puerto de telnet (--destport 23) de mi router ADSL, y ahora me es imposible conectar con el telnet del router.

¿Significa esto que yo, que precisamente estoy explicando estas cosas, tengo un grave problema de seguridad? Realmente no, por tres motivos. En primer lugar, porque el puerto de Telnet lo tengo abierto sólo hacia mi red local, por lo que sólo podría atacarme... yo mismo. En segundo lugar, porque no es un servicio de importancia crítica, es decir, me da igual tirarme el tiempo que sea sin poder acceder al telnet de mi router, ya que sólo lo uso muy rara vez, cuando tengo que modificar algo en la configuración. En tercer lugar, al tratarse de un router hardware y no de un simple programa de PC, tendría que esperar a que saliese una nueva actualización del firmware que solucionase este problema, así que en cualquier caso no está en mi mano la solución, si no en la del fabricante del router. Ya que la cosa se está calentando un poco,

vamos a probar alguna técnica más de hacking relacionada con TCP.

Ataques por adivinación de número de secuencia con Hping2.

Vamos ahora con una técnica realmente interesante que, de hecho, utilizó incluso el propio Kevin Mitnick (uno de los hackers más famosos de la historia) como parte de las andanzas que le hicieron terminar en la cárcel (aplicaos el cuento, jeje).

En este caso, no se trata de un simple ataque DoS, como el SYN Flood, si no de un ataque mucho más versátil que nos permitirá **colarnos en conexiones ajenas**, con todo lo que ello implica.

Conseguir un ataque de este tipo con éxito es realmente complicado, así que lo que voy a contar, que en la teoría puede parecer tan "sencillo", en la práctica choca con mil y un inconvenientes, empezando por la dificultad que comenté antes de hacer un IP Spoofing sin que se enteren los routers que transportan el paquete.

Como lo importante es comprender la teoría de las cosas, y no meternos en líos, voy a explicar las bases de este tipo de ataques.

Para comprender el funcionamiento de estos ataques debemos recordar qué es lo que define exactamente una conexión, es decir, lo que identifica unívocamente a una conexión para diferenciarla de cualquier otra de Internet. Pues son estos los parámetros: **una IP de origen, una IP de destino, un puerto de origen, un puerto de destino, y los números de secuencia de cada una de las dos partes.**

Por tanto, si conociéramos todos estos datos, teniendo una herramienta como Hping2 que nos permite crear paquetes a medida, podríamos insertar cualquier dato en una conexión ajena.

Por ejemplo, si sabemos que la máquina A, con IP 192.168.1.2, tiene establecida una conexión de FTP (puerto de destino 21) con la máquina B, con IP 192.168.1.5, utilizando como puerto de origen el 3560, nos bastaría con saber el número de secuencia que está utilizando la máquina A para poder inyectar paquetes en su conexión de FTP. Supongamos que sabemos que su número de secuencia es el 24560.

Bastará con hacer:

hping2 192.168.1.5 --spoof 192.168.1.2 --baseport 3560 --destport 21 --tcpseq 24560 --file comandos.txt --data 14 --count 1

Con esto enviamos un único paquete (--count 1) enviando como IP spoofeada la de la máquina A (--spoof 192.168.1.2), e inyectando como datos unos comandos de FTP que hemos metido previamente en el archivo COMANDOS.TXT.

Por supuesto, el gran problema de esto es que es realmente complicado conocer el número de secuencia de una conexión ya que, además de ser un número realmente grande (32 bits), va cambiando constantemente a lo largo de una conexión.

Hping2 nos ofrece una herramienta para ayudarnos a la hora de adivinar el número de secuencia, comprobando si un determinado sistema utiliza números de secuencia fáciles de predecir. Para ello nos da la opción **--seqnum**, que hace un análisis de los números de secuencia utilizados por un sistema:

hping2 192.168.1.5 --seqnum --destport 21 --syn

Con esto veríamos cómo varían los números de secuencia de la máquina B cada vez que intentamos conectar con su puerto de FTP. Hping2 nos mostrará el número de secuencia utilizado, y el incremento con respecto al utilizado anteriormente. Si este incremento es siempre el mismo, entonces estaremos ante una máquina con números de secuencia "fácilmente" predecibles.

Una vez que ya tenemos una idea del rango de números de secuencia que puede utilizar la máquina que queremos suplantar, podemos intentar lanzar miles de paquetes iguales, en los cuales sólo cambie el número de secuencia, y esperar que el azar nos recompense con la suerte de que alguno de ellos haya acertado, y el paquete se inyecte correctamente en la conexión ajena.

6.1. Nemesis para Windows

Para empezar, os recuerdo que en el directorio de Nemesis tenéis un archivo de ayuda para cada protocolo. En este caso el que nos interesa es el archivo **nemesis-tcp.txt**. Como ya me estoy quedando sin espacio, os resumo brevemente las opciones que nos da Nemesis para TCP, que son bastantes:

-x : permite especificar el **puerto de origen**.
-y : permite especificar el **puerto de destino**.
-s : permite especificar el **número de Secuencia**.
-a : permite especificar el **numero de confirmación**.
-fS : activa el flag **SYN**
-fA : activa el flag **ACK**
-fR : activa el flag **RST**
-fP : activa el flag **PSH**
-fF : activa el flag **FIN**
-fU : activa el flag **URG**
-w : permite especificar el **tamaño de la ventana**.
-u : permite especificar el campo **puntero de urgencia**.
-o : permite incluir un fichero que contenga las **opciones TCP** que queramos.
-v : activa el modo **verbose** que nos da información más detallada de lo que estamos haciendo.

Os recuerdo también que necesitaremos un par de opciones referentes a IP:

-D : permite especificar la **IP de destino** (imprescindible).
-S : permite especificar la **IP de origen** (IP Spoofing).

Por ejemplo, si queremos enviar un paquete de solicitud de conexión al FTP de Rediris podemos hacer:

Nemesis tcp -v -S 192.168.1.1 -D 130.206.1.5 -x 1000 -y 21 -fS -a 0

En primer lugar especificamos nuestra IP (-S 192.168.1.1), que en este caso es una IP de red local porque nos encontramos detrás de un router ADSL. En segundo lugar indicamos la IP de destino, es decir, la del servidor FTP de Rediris (-D 130.206.1.5). A continuación especificamos los puertos de origen y de destino (-x 1000 -y 21). A continuación, activamos el flag SYN para este paquete (-fS). Por último, ponemos un 0 en el campo número de confirmación, ya que es para una conexión aún no establecida (-a 0).

IDS (SISTEMA DE DETECCION DE INTRUSOS) TERCERA ENTREGA

PLUG-INS DE SALIDA EN SNORT

Tercer y último artículo de la "serie" IDS... y no me "gustaban" las dedicatorias... los dos artículos anteriores también fueron dedicados a... éste no podría ser menos... en este caso no es a una persona en concreto, este va dedicado a un grupo, un grupo formidable y con todo mi reconocimiento...ese grupo sois vosotros, los lectores de esta revista....

Para tod@s vosotros

Sin más preámbulos, empecemos!!!

Si no utilizamos *plug-ins* de salida, ***snort*** volcará los resultados de alertas y *logs* en archivos de texto plano dentro del directorio por defecto o en el los archivos indicados que definen las reglas o directivas del *pre-procesador.*

Esos archivos, aunque comprensibles y legibles, son tediosos y difíciles de seguir, la intención que buscan los plugins de salida es la de facilitar la lectura de los mismos, ubicar los resultados en Bases de Datos, visualizar en tiempo real (o diferido) esos resultados en formatos *"mas amigables"* y fáciles de seguir, como pueden ser documentos *HTML, XML*, gráficos estadísticos y otros...

Desde luego la forma más eficiente de que ***snort*** trabaje no es precisamente el uso de *plug-ins* muy pesados, los análisis en tiempo real y registro de alertas en bases de datos consumen más recursos y utilizan un ancho de banda más intensos, además una base de datos para que sea efectiva ha de mantenerse, lo que implica también costes económicos y recursos humanos.

También se necesitan aplicaciones *"extra"* para manejar adecuadamente esos *plugins*, desde Gestores de Bases de Datos tipo *MySql*, intérpretes de *perl, XML*, lenguajes de programación como *php*, etc...

No todo son desventajas, por el contrario, los pros de utilizar este tipo de *plugins* son: Recibir una información más acertada en tiempo real, registro correlativo en bases de datos con la consiguiente mejora de filtrar por alertas, contenidos, tipos de intrusiones, etc.. en resumen, la principal ventaja reside en la flexibilidad del análisis, la escalabilidad y una mejor presentación de los resultados.

Empecemos con uno sencillito, independiente de la plataforma, de libre distribución, etc... tanto para *LiNUX* como para *Windows* utiliza los mismos links, es en un *bytecode*... por eso *Java* es multiplataforma

Se llama ***SAM (Snort Monitor Alert)*** y lo podréis encontrar en:

http://aleron.dl.sourceforge.net/sourceforge/snortalertmon/sam_2002-08-26_binary.zip

Tienes una versión mucho más reciente en (no *testeada*):

http://aleron.dl.sourceforge.net/sourceforge/snortalertmon/sam_20040323_bin.zip

Necesitaremos unas cuantas cosas para poder usar este *plugin*, si hiciste *"tus deberes"* en los números anteriores ya deberías tener instalado varias de estas cosas, si no ahora es el momento, porque para ***SAM*** y para los otros que nos vienen por delante los vamos a necesitar:

Gestor de Base de Datos (puede ser *MySQL, ORACLE, PostGreSQL...*) SAM sólo puede usar MySQL.

Java Runtime Environmet, esto es la máquina virtual de java para poder ejecutar los ***archivos .jar***

Tanto para *LiNUX* como para *Windows* puedes descargar ***Mysql*** desde aquí:

http://www.mysql.com/downloads/index.html

No tienes más que elegir la plataforma deseada, muchas distribuciones de *LiNUX* ya incluyen *mysql*, si es así no será preciso descargarse nada a menos que quieras actualizar la revisión.

Actualmente se está desarrollando y probando la versión 5.0 pero es mejor que usemos la 4.0

Configurando SAM para Windows

La instalación de ***mysql*** en *Windows* es muy simple, lo de siempre....ejecutamos el instalador y seguimos los pasos... si quieres seguir los ejemplos que vienen a continuación *"al pie de la letra"* **recuerda** que el directorio de instalación de *mysql* será ***C:\mysql***.

En la Revista número 8 cuando se hablaba de la creación de un foro phpBB2 se explica más detalladamente los pasos, aquí sólo haré referencia a lo mínimo indispensable...

Tras finalizar la instalación, buscaremos dentro de la ***carpeta mysql/bin*** un archivo llamado ***winmysqladmin.exe***



Pantalla 1. Ubicación del archivo winmysqladmin.exe

Lo ejecutamos y, si es la primera vez que lo haces, te pedirá un nombre de usuario y contraseña para el acceso al Gestor.



Pantalla 2. Usuario y Contraseña para administrar MySQL

Puedes poner cualquier cosa.... pero recuerda lo que pones

Ahora verás en la bandeja del sistema un nuevo icono... el ***semáforo***....

Si no ves el semáforo, vuelve a ejecutar el archivo ***winmysqladmin.exe*** como antes... y por supuesto **la "*luz*" del semáforo tiene que estar verde...**

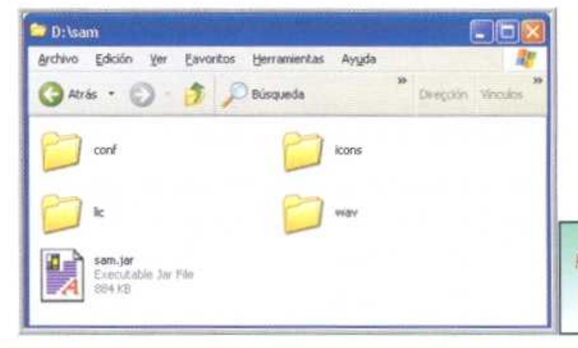
Ahora ***instalemos Java Runtime Environment***

http://java.sun.com/j2se/1.3/jre/ (al igual que antes eliges la plataforma y descargas los binarios correspondientes).

Y la instalación en *Windows*, como todas.... siguiente, siguiente.... finalizar y por si *"las moscas pican"*, reiniciamos....

Ahora vamos a descomprimir ***SAM***, en mi PC lo guardé ***en D:\sam***, tu eres libre de decidir dónde.

Tras descomprimirlo en la carpeta deseada verás esto:



Pantalla 3. Ubicación del archivo sam.jar

Antes de ejecutar nada, hay que seguir unos cuantos pasos...

Configurar el archivo sam.properties que está en la carpeta ***d:/sam/conf***

Crear la base de datos y tablas necesarias para el ***plugin database***

Modificar el archivo de configuración de ***snort*** para que utilice *mysql*

Si accedemos a la ***carpeta conf de SAM*** encontraremos un archivo y otra carpeta, de lo que se trata es de editar el archivo ***sam.propiertes*** y darle los valores que queramos, el contenido original es este:



sam.properties - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
email.host=smtp.your-domain.com
email.from=snort@your-domain.com
email.to=your-email@your-domain.com
email.active=false

alertlevel.medium=30
alertlevel.high=40

mainpanel.refresh=3
```

Pantalla 4. Contenido del archivo sam.propiertes

Nos permitirá **enviar alertas mediante el uso del correo electrónico**, para ello has de poner el servidor de correo saliente que uses (***smtp***), la cabecera ***from***(quien envía el mensaje), la cabecera ***to*** (la dirección mail destino) y en la línea ***email.active=*** cambiar el valor de ***false*** por ***true***, en caso contrario no se enviará nada aunque configuremos adecuadamente.

Las líneas ***alertlevel.high y alertlevel.medium*** son variables que usará ***SAM*** para enviar alertas, realmente ***SAM*** sólo envía las alertas *"de código rojo"* cada cinco minutos, esto lo entenderás más adelante y la línea ***mainpanel.refresh=3*** simboliza el los minutos que deben transcurrir para que ***SAM*** actualice el panel de control... en un momentito lo entenderás todo.

Tampoco te preocupes mucho por las variables ***alertlevel*** puesto que todavía no están implementadas....

Por ejemplo, podemos configurarlo así:



sam.properties - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
email.host=mail.forohxc.com
email.from=vic_thor@forohxc.com
email.to=vic_thor@forohxc.com
email.active=true

alertlevel.medium=30
alertlevel.high=40

mainpanel.refresh=1
```

Pantalla 5. Nuevo contenido del archivo sam.properties

Esta configuración enviaría las alertas de código rojo a mi dirección mail, usando el servidor smtp mail.forohxc.com y actualiza el panel de administración cada minuto... jeje, esas cuentas de correo NO SON MIAS, son ficticias, no vayas a enviarme mails allí que no me llegarán.

Bueno, pues guardamos ese archivo (***te recomiendo que no uses email.active a true para empezar, para evitar problemas "de principiante")***

***Ahora vamos a* crear la base de datos** que necesitamos, sus tablas y sus campos...

Uffff, tarea de titanes... pero afortunadamente tenemos un ***source*** para *mysql* que lo hará solito... lo que pasa es que no sabes donde está... pero lo tienes

Vamos a situarnos en la ***carpeta c:\snort\contrib*** (¿recuerdas que era una carpeta para *"contribuciones"*?, pues esta es una de ellas que nos facilitará la vida).



Pantalla 6. Ubicación del archivo create_mysql

El archivo que nos interesa es ***create_mysql***



Copiamos ese archivo a la carpeta ***C:\mysql\bin***



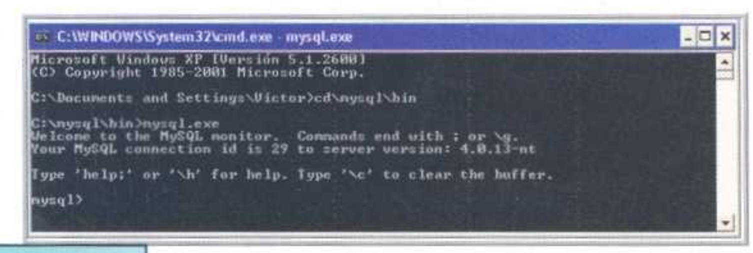
Pantalla 7. Nueva ubicación del archivo create_mysql

Con esto nos evitaremos tener que escribir la ruta cuando le pidamos a *mysql* que genere la Base de datos necesaria.

¿Y si no tengo ese script? Pues lo puedes descargar de:

http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/snort/snort/contrib

Ahora abrimos una línea de comandos (***inicio-ejecutar-cmd***) y **cambiamos** al directorio ***c:\mysql\bin* y ejecutamos la orden *mysql.exe***



Pantalla 8. Ejecutando el SDGBD

NO TE CONFUNDAS... ahora estamos ante el gestor de *mysql* y NO en la *shell* de Windows. Puedes ver en la imagen que en la última línea sale ***mysql>***, todo comando que introduzcamos a partir de ahora deberá ser un comando de mysql (ya no sirven los comandos de DOS).

Si no sabes nada de *SQL* no importa, tampoco es muy complejo lo que vamos a realizar y está guiado *"paso a paso"* pero si quieres un buen curso de *SQL* y **GRATIS**... pasa a nuestros foros:

Nuestro compañero ***Yorkshire*** nos ha dejado dos joyas en la sección de FAQ,

Curso de SQL:
http://www.hackxcrack.com/phpBB2/viewtopic.php?t=12222

Curso de PL/SQL:
http://www.hackxcrack.com/phpBB2/viewtopic.php?t=13065

Bueno, lo que necesitamos es crear una Base de datos, seleccionarla y crear la estructura de tablas y campos, **vamos a elegir como nombre *idstest*** y la creamos y seleccionamos así:

```
C:\WINDOWS\System32\cmd.exe - mysql.exe
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 29 to server version: 4.0.13-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database idstest;
Query OK, 1 row affected (0.48 sec)

mysql> use idstest;
Database changed
mysql> _
```

Pantalla 9. Crear la Base de Datos

Ahora toca lo *difícil*... tendríamos que conocer la estructura que manejará ***snort***, las tablas, sus nombres, los campos, sus tipos, longitudes, etc... pero para eso está el archivito que copiamos anteriormente al *directorio /bin* de mysql... ese que se llamaba ***create_mysql***, hacemos esto:

source create_mysql

```
C:\WINDOWS\System32\cmd.exe - mysql.exe
mysql> use idstest;
Database changed
mysql> source create_mysql;
Query OK, 0 rows affected (0.09 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 0 rows affected (0.02 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.02 sec)
```

Pantalla 10. Creación de la estructura mediante el script source create_mysql

No puse la salida completa, para que la pantalla no sea excesivamente grande, pero resumiendo, ***la orden source create_myql; creará la estructura que necesitamos...***

Claro que **mejor sería asignar un nombre de usuario y password** para que no entre el primero que llega, ... **mejor sigue el curso de SQL que nos proporcionó yorkshire...** aunque cuando veamos el siguiente *plugin* te enseñaré como hacerlo

Ahora que ya lo tenemos todo preparadito, nos falta explicarle a ***snort*** que utilice como ***plugin* de salida la base de datos que nos acabamos de crear**...

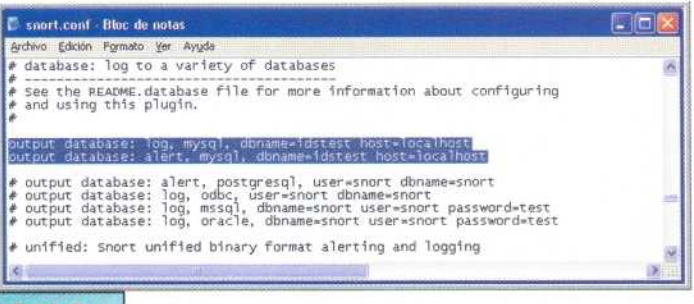


En artículos...

En artículos anteriores hemos utilizado distintos archivos de configuración, como imagino que ya habrás hecho muchas prácticas, inclusión de *preprocesadores*, etc... para este ejemplo y los que vienen usaremos el ORIGINAL, es decir el archivo snort.conf, así evitaremos problemas de "*nomenclaturas*".

Ese archivo está alojado en la carpeta C:\snort\etc\snort.conf, lo editamos, buscamos la **sección correspondiente** (si tienes dudas mira la imagen siguiente) y **añadiremos estas dos líneas**:

output database: log, mysql, dbname=idstest host=localhost
output database: alert, mysql, dbname=idstest host=localhost



```
snort.conf - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
# database: log to a variety of databases
# ----------------------------------------
# See the README.database file for more information about configuring
# and using this plugin.
#

output database: log, mysql, dbname=idstest host=localhost
output database: alert, mysql, dbname=idstest host=localhost

# output database: alert, postgresql, user=snort dbname=snort
# output database: log, odbc, user=snort dbname=snort
# output database: log, mssql, dbname=snort user=snort password=test
# output database: log, oracle, dbname=snort user=snort password=test

# unified: Snort unified binary format alerting and logging
```

Pantalla 11. Modificación del archivo snort.conf, sección 3, plugin database

Si hubiésemos asignado nombre de usuario y contraseña a la base de datos tendríamos que haber escrito:

output database: log,mysql,dbname=idstest user=usuario password=contraseña host=localhost

output database: alert,mysql,dbname=idstest user=usuario password=contraseña host=localhost

Donde **usuario y contraseña será** el nombre y *password* que hubiésemos asignado. Una vez modificado el archivo de configuración, lo guardamos...

Ahora **vamos a iniciar *snort*** desde la línea de comandos como va siendo habitual...


Pantalla 12. Ejecución de snort



```
C:\WINDOWS\System32\cmd.exe
C:\>cd \snort\bin

C:\Snort\bin>snort -de -A fast -l C:\snort\log -c C:\snort\etc\snort.conf_
```

Y quedará esperando como siempre...

```
C:\WINDOWS\System32\cmd.exe - snort -de -A fast -l C:\snort\log -c C:\snort\etc\snort.conf
-*> Snort! <*-
Version 2.1.1-ODBC-MySQL-FlexRESP-WIN32 (Build 24)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)
1.7-WIN32 Port By Michael Davis (mike@datanerds.net, www.datanerds.net/~mike)
1.8 - 2.1 WIN32 Port By Chris Reid (chris.reid@codecraftconsultants.com)
```

Pantalla 13. Ejecución de snort... capturando paquetes...

Nos toca ***ejecutar SAM***... para ello accedemos al directorio donde está instalado (recuerda que era ***D:\sam***) y podemos hacerlo de dos modos:

Haciendo **doble clic en el archivo *sam.jar*** desde el *interface* gráfico

Desde una **línea de comandos: *java –jar sam.jar***

Pantalla 14. Acceso a la base de datos

El caso es que de una u otra forma nos aparecerá esto:

Database Login
Database: MySQL
Hostname: 172.28.0.50
Database Name: idstest
Username:
Password:
OK | Cancel

Escribiremos la dirección IP del equipo y el nombre de la base de datos que creamos en mysql, o la dirección 127.0.0.1 si se trata del mismo equipo donde están ejecutando tanto *mysql* como *SAM*

Lógicamente daremos el nombre de usuario y contraseña si los hubiésemos creado, OK y......

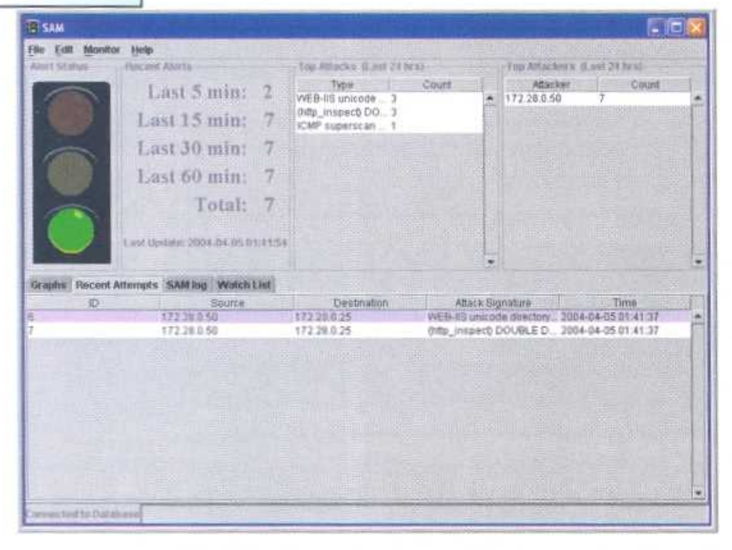
Pantalla 15. Alertas y control en tiempo real

jeje, si tienes los altavoces conectados oirás una "voz profunda"... simpático, no?

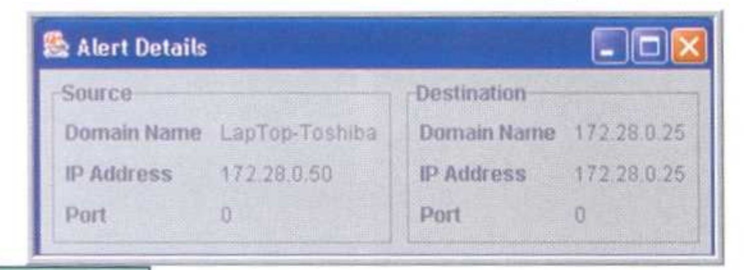
Bueno en el ejemplo de arriba, ya *"trastee"* un poco con la red... *puedes ver en la imagen anterior que intenté 3 accesos por unicode a un webserver, unos cuantos escaneos de puertos, etc...*

Podemos ver algo más de los ataques si usamos la ***ficha de Recent Attemps***, pero antes de que pasen 5 minutos o se borran...

Pantalla 16. Alertas y control en tiempo real. Ficha Ataques Recientes



Y si seleccionamos una de ellas y pulsamos ***botón-derecho show details***, aparece esto:



Pantalla 17.Detalle de las direcciones del ataques

SAM es muy fácil de utilizarse, no tiene ningún misterio, el resto te lo dejo a ti...

Configuración e instalación de SAM para LiNUX

No voy a repetir lo mismo, la instalación para *LiNUX* es similar y prácticamente todos los pasos descritos para *Windows* se resuelven del mismo modo, recuerda que la estructura de directorios donde se instaló ***snort*** en plataformas *LiNUX* difiere de las de *Windows*.

Es importante que hayas instalado convenientemente ***snort***, recuerda que hay binarios y fuentes que no dan soporte a Bases de Datos y que si deseamos usar *MySQL* u otros Sistemas Gestores de Bases de Datos es necesario generar los archivos de instalación con el ***script ./configure*** usando las opciones ***-with-mysql=DIR*** como lo hicimos el mes pasado.

Por otro lado, la instalación de *mysql* es diferente aunque muy probablemente ya la tengas en la distribución que utilices y si no.... a bajarla e instalarla y ***no se te olvide crear la base de datos y utilizar el script create_mysql como*** hemos hecho con la versión de *Microsoft*.

Para que puedas usar el ***script*** debes haberte bajado el ***código fuente de snort***, puesto que en las versiones RPM puede no encontrarse la carpeta ***contrib***.

En lo que se refiere a ***Java Runtime Environment***, más de lo mismo, lo más probable es que lo tengas instalado, por lo que realmente en plataformas *LiNUX* no será preciso más que descargar el módulo ***SAM*** y **configurar los archivos *sam.properties* y *snort.conf* IGUAL** que hemos descrito para *Windows*.

Brevemente describimos los pasos para la instalación de SAM para LiNUX

Suponemos que las librerías *libpcap y libnet* ya están instaladas, si no sabes como hacerlo revisa el artículo del mes pasado). También se supone que disponemos de *MySQL* instalado.

1º) Instalación de *snort* con soporte *Mysql* u otros...

Descomprimir el código fuente de snort en un directorio cualquiera, por ejemplo en /root/Taller_snort/temp/snort-2.1.1

```
cd /root/Taller_snort/temp/snort-2.1.1
./configure -enable-smb-alerts -enable-flexresp -with-mysql
make
make install
```

2º) **Instalar Java Runtime Environment** descargándolo desde la web de *Sun*.

3º) **Descargar SAM y** Modificar el archivo ***sam.properties*** como hicimos para *Windows*

4º) Modificar el archivo ***snort.conf*** como se explicó en la instalación de ***SAM*** para Windows.

5º) **Iniciar el demonio mysql** si no está iniciado anteriormente ***mysqld***

6º) **Copiar el archivo *create_mysql*** al directorio de trabajo y crear la base de datos:

```
mysql
create database idstest;
use idstest;
source create_mysql;
exit
```

7º) **Ejecutar *snort*** desde la línea de comandos de terminal

8º) **Ejecutar SAM**

Si te digo la verdad, nunca usé ***SAM*** con *LiNUX*, siendo más sincero... nunca usé *SAM "en serio"*. El motivo de incluirlo en este artículo es por su sencillez y como aperitivo de lo que viene a continuación

Son muchos los *plugins* existentes para ***snort***, nos perderíamos en los detalles y profusión de ellos, en la web www.snort.org encontrarás experiencias y opiniones de otros muchos, pero si hay uno especialmente utilizado y bien desarrollado es ***ACID***, quizás el más usado por los administradores que usan ***snort***.

Antes de comenzar con él te pongo una tabla de otros añadidos con los que puedes investigar:

Plug-in	Link de acceso	Descripción del plug-in
SnortSnarf	www.silicondefense.com/software/snortsnarf	Diagnósticos en formato HTML
Snortplot.php	www.snort.org/dl/contrib/data_analysis/snortplot.pl	Script en perl para el trazado de intrusiones
Swatch	http://swatch.sourceforge.net	Monitor en tiempo real de sucesos que incluye alertas via e-mail
ACID	http://acidlab.sourceforge.net	Análisis y resguardo de logs en Bases de datos, necesita PHP, Apache y plugins snort para acceso a BBDD
Incident.pl	www.cse.fau.edu/~valankar/incident	Script en perl para crear informes en archivos logs
Loghog	http://sourceforge.net/projects/loghog	Analizador de logs que puede generar alertas vía e-mail, bloquear tráfico interactuando con las reglas definidas en IpTables
Oinkmaster	http://oinkmaster.sourceforge.net	Utilidad para el mantenimiento de reglas
SneakyMan	http://sneak.sourceforge.net	Configurador de reglas basado en GNOME
SnortReport	www.circuitsmaximus.com/download.html	Módulo que genera informes en tiempo real de intrusiones

Ahora a por ***ACID***....

Configuración e Instalación de ACID

Aunque podemos utilizar ***ACID*** en versiones *Windows* o *LiNUX*, en esta ocasión voy a centrarme en los **pasos de instalación de ACID en *LiNUX*** y dejaremos los pasos de instalación para *Windows* en una forma *"menos detallada"*

Analysis Console for Intrusión Databases (ACID) es un *plugin* que interactúa con *scripts* ***php***, ***servidores web y bases de datos*** y es muy probable que en la misma distribución de ***snort*** ya tengas incluido los fuentes de ***ACID***, como es el ejemplo que nos ocupa... cuando descargamos el código fuente de ***snort***, al descomprimir el archivo *tarball* al directorio de trabajo, se creó una carpeta llamada ***contrib*** dentro de la cual está ***ACID***

Las principales características de ACID, son:

Interface gráfico para buscar y consultar las alertas y *logs* de la base de datos
Un buscador para decodificar y buscar paquetes de datos en capa 3 y 4
Organizar alertas por grupos
Enviar *mails* y exportar a bases de datos
Generación de gráficos

Los requisitos para instalar ACID, son:

Sistema Operativo *LiNUX* con *Xwindow* o S.O. *MS-Windows*
Uno o varios Servidores Web en nuestra red, mejor ***Apache***
Lenguaje ***PHP*** y configurar el webserver para usar este lenguaje en sus páginas
Uno o varios Servidores de Bases de Datos, ***MySQL***, ORACLE, PostgreSQL, etc.
Algunas ***librerías gráficas*** y para archivos comprimidos
Plug-ins para PHP como ***PHPplot*** o librerías ***JPGraph***
Disponer de ***snort*** instalado y **configurado para usar Bases de Datos**

Y este artículo

Buaaahhhhh!!!! Qué de cosas... ¿las tengo? ¿donde?

Bien, vayamos por partes...

Si has seguido todos los artículos y has instalado todo lo que llevamos hasta hoy, ya debes disponer de ***Snort y MySQL*** y lo normal es que tu distribución ya incluya *PHP y Apache* como aplicaciones integradas en la misma.

Luego sólo nos harán falta las librerías, pero por si acaso voy a realizar una muy breve explicación de cómo obtener e instalar ***Apache*** y ***PHP***, así como de cómo **configurar ambos para que las páginas *php* se puedan servir desde Apache.**

Como cada uno tiene una distribución y cada cual dispondrá de unos paquetes u otros instalados, voy a suponer *"que no tenemos NADA"* y vamos a ir instalando todo...

Aunque estamos en la sección de *LiNUX*, si estás usando *Windows*, descarga los programas que voy indicando para la plataforma Win32, así cuando le toque el turno a *Windows* ya tendrás las herramientas necesarias.

Escenario de Instalación

Sistema NIDS. Snort 2.1.1

-- Máquina 172.28.0.200
-- Soporte para Bases de Datos, Respuestas Flexibles
-- Archivo de configuración: /etc/snort/snort.conf
-- Directorio de reglas: /etc/snort/rules
-- Directorio de logs: /var/mis_logs
-- Librerías y dependencias: libpcap y libnet

Servidor Web. Apache 1.3.27

-- Máquina: 172.28.0.200
-- Carpeta de documentos: /www/htdocs
-- Archivo de configuración: /www/conf/httpd.conf
-- Modulos php y/o Aplicaciones CGI configurados

Servidor de Bases de Datos MySQL 4.x

-- Máquina: 172.28.0.200
-- Base de datos de alertas para snort: testids

Lenguaje PHP4

-- Instalado en la máquina: 172.28.0.200
-- Configurado con soporte para MySQL, GD y sockets

Otras librerías necesarias

-- Máquina: 172.28.0.200
-- ADODB, para comunicar PHP con MySQL

-- GD, librerías para gráficos GIF/JPEG/PNG
-- GD depende de libpng, libjpeg y zlib
-- Phplot, scripts en PHP para construcción de gráficas estadísticas
-- JPGraph, Como phplot y dependerá de la versión de ACID que instalemos

Servidor Web. Apache 1.3.27

-- Plug-in ACID para snort
-- Máquina: 172.28.0.200

Como ves **TODO** deberá estar instalado y configurado sobre la misma máquina, pero no tiene por qué ser así... podríamos perfectamente disponer de un *Servidor Web* independiente del *IDS* y del servidor *MySQL*, es decir, 3 máquinas diferentes...

Este es el escenario más sencillo para explicarte como funciona ***snort+acid+mysql+php+apache*** en este artículo, de otra forma nos perderíamos bastante en los detalles de cada uno, a parte que para seguir esta práctica necesitarías disponer de 4 ó 5 máquinas diferentes... *y ya sé que pocos tenemos tantas cosas en casa... algunos locos si*

*Aunque bien pensado... siempre nos queda **vmWare** o **Virtual PC***

Instalar Apache con módulos dinámicos compartidos (DSO)

Podemos encontrar apache para todas las plataformas y distribuciones en:

http://www.apache.org

Supongamos que bajamos el archivo ***apache-1.3.29.tar.gz*** desde aquí:

http://apache.rediris.es/httpd/apache_1.3.29.tar.gz

Ya... ya sé que andamos por la 2.x pero es que esta revisión es la que tenía por un CD y así no me toca bajar más paquetes... que menuda sesión que llevo si quieres baja la última, y recuerda cambiar los nombres que pongo a continuación:

*Vamos a instalar esa versión de **Apache con soporte para PHP4 como módulo del servidor Web.***

*Descomprimimos el archivo **apache_1.3.29.tar.gz** en un directorio... el mío es:*

/root/Taller_snort/apache_1.3.29

Lo que pretendemos es instalar el webserver en /www de tal modo que la estructura quede más o menos así:

*-- Directorio del archivo de configuración: **/www/conf/httpd.conf***
*-- Directorio de ejecución: **/www/bin/httpd***
*-- Directorio para alojar las páginas web: **/www/htdocs***
*-- Directorio de módulos: **/www/modules***
*-- Directorio para la definición de módulos dinámicos: **/www/bin/apxs***

Hay más pero estos son los que nos interesan por el momento...

*Estas líneas instalarán apache en el directorio **/www** y creará los otros....*

```
cd /Taller_snort/apache_1.3.29 (cambiamos al directorio donde están los fuentes)
./configure --prefix=/www --enable-module=so
make
make install
```

Ahora comprobemos si está habilitado la opción de módulo dinámico

/www/bin/httpd -l

*Y deberíamos ver varios módulos instalados, el que DEBE estar es: **mod_so.c** si no es así no podremos seguir, pero debe ser así*

***NO TOQUES NADA más**, el resto lo haremos cuando instalemos y configuremos PHP*

Puedes probar que funciona, eso sí...

www/bin/httpd start

Y ahora abre el navegador web que uses y accede a 127.0.0.1 deberías ver la web "por defecto"

Paramos el servidor web:

/www/bin/httpd stop

Instalando PHP4 como módulo de Apache

Necesitamos instalar ***PHP*** con soporte de Bases de Datos ***MySQL***, Librerías Gráficas (GD support) y ***Socket Support***

Nos descargamos **las fuentes de php en** http://www.php.net/downloads.php

Para ***Linux*** http://www.php.net/get/php-4.3.5.tar.gz/from/a/mirror

Para ***Windows*** http://www.php.net/get/php-4.3.5-Win32.zip/from/a/mirror

Descomprimes el archivo php-4.3.5.tar.gz *en el directorio de trabajo que quieras... en mi caso es* ***/root/Taller_snort/php-4.3.5***

cd /root/Taller_snort/php-4.3.5
./configure --wtih-mysql --with-gd --with-zlib --enable-sockets --with-apxs=/www/bin/apxs

Si ***mysql*** *o las* ***librerías GD, zlib...*** *no están instaladas o no las encuentra, hay que instalarlas primero o indicar el directorio donde están, p.e.* ***-with-mysql=/usr/local/mysql***

Si las ***librerías GD, zlib no las tienes instaladas****, en la próxima sección te explico como se instalan, las colocas por ejemplo en* ***/usr/local/lib/gd1.3 y /usr/local/lib/z-lib-1.1.3*** *y ejecutas el script* ***./configure*** *apuntando a esos directorios, por ejemplo:*

./configure --wtih-mysql --enable-sockets --with-apxs=/www/bin/apxs \ --with-gd=/usr/local/lib/gd1 --with-zlib=/usr/local/lib/z-lib-1.1.3



IMPORTANTE

IMPORTANTE: Si ya tienes apache instalado y comprobado que acepta la configuracón de módulos dinámicos (DSO) mediante httpd –l pero lo tienes en otro directorio, recuerda cambiar la opción --with-apxs=/[directorio donde tengas el archivo apxs] Normalmente ese archivo está dentro del directorio /bin de la instalación.

Luego, lo mismo de siempre: ***make y make install***

Ahora tenemos que "tocar" el archivo ***httpd.conf*** *de* ***apache***

gedit /www/conf/httpd.conf *(o utiliza otro editor de texto diferente a* ***gedit****) y añade estas líneas en la sección* ***Addtype*** *si es que no las tiene:*

AddType application/x-httpd-php .php .phps .php3 .phtml .php4

Verifica *que tienes una línea dentro de la sección Dynamic Shared Object (DSO) Support,*

LoadModule php4_module libexec/libphp4.so

También sería bueno cambiar la línea ***Servername*** *y ponerle la IP del host (****172.28.0.200****) y después:* ***Guardamos los cambios.***

Para probar que todo fue bien, créate un archivo de texto con este contenido:

```
<?
phpinfo();
?>
```

Y lo guardas en ***/www/htdocs*** *con el nombre* ***index.php*** *(por ejemplo)*

Iniciamos de nuevo apache:
/www/bin/httpd start

Ahora navega a la dirección http://172.28.0.200/index.php *y deberías ver la página php que ofrece la función* ***phpinfo();***

La gente de *Windows*.... no debe hacer eso... al menos no así, ya les llegará su hora.

Instalando las librerías necesarias

Librerías gráficas y compresión de archivos (necesarias pero seguramente incluidas)

Lo más probable es que ya dispongas de las librerías y dependencias para archivos gráficos del tipo **jpeg, png o gif...** si no es así puedes bajar los fuentes, los rpm o los binarios para *LiNUX* en:

http://www.boutell.com/gd/ (no disponible en *Windows*, pero no te preocupes... nuestro amado *Windows* ya es capaz de manejar esos formatos de archivos *"de fábrica"*)

Bueno *LiNUX* también, pero por si acaso....

GD, necesita además otras dependencias:

Librerías **libpng**: http://www.libpng.org/pub/png/libpng.html
Librerías **libjpeg:** http://www.ijg.org/
Librerías **zlib:** http://www.gzip.org/zlib/

Como ya he dicho es muy probable que no necesites nada de eso y que tanto *Windows* como la distribución de *LiNUX* que dispongas ya las tenga instaladas.

La única librería que merece explicar *"algo"* sobre su instalación es ***libpng***... yo me descargué la que en el momento de escribir este texto la versión 1.2.5 y el proceso de instalación es:

Descomprimes el archivo tar en el directorio que quieras, por ejemplo ***/root/Taller_snort/Sources/linpng-1.2.5***

Una vez extraídos los archivos dentro de ese directorio se habrán creado varios archivos y otros directorios, busca y accede al directorio ***scripts***:

```
cd ./scripts
ls
```

Y saldrán varios ***makefiles.???*** Has de elegir el que corresponda a tu distribución o sistema operativo, en mi caso es ***makefile.linux***, lo copiamos al directorio anterior y le damos el nombre ***makefile***

```
cd ..
cp ./scripts/makefile.linux
mv makefile.linux makefile
```

Esto es porque estas librerías no disponemos del script ./configure, ya vienen los ***makefiles*** *"hechos"* por lo que sólo nos restará una vez copiado y cambiado el nombre crearlos con ***make***

```
make
```

Librerías phplot o JpGraph

Necesarias si quieres utilizar **ACID** para generar gráficas estadísticas....

Phplot realmente no es una librería, ***es un script php***, de momento lo bajamos y luego lo instalamos:

http://aleron.dl.sourceforge.net/sourceforge/phplot/phplot-5.0rc1.tar.bz2

Librerías JPGraph

Las últimas versiones de ***ACID*** precisan de estos scripts PHP en lugar de los phplot, tanto unos como otros son programas PHP y tanto los que uséis *Windows* o *LiNUX* tendréis que descargarlos si queremos usar ***ACID*** con herramientas gráficas.

http://www.aditus.nu/jpgraph/jpdownload.php

Librerías ADODB

Esta si que es **IMPRESCINDIBLE**, también **son programas en PHP,** no librerías... y los puedes descargar en:

http://php.weblogs.com/adodb#downloads

Si por cualquier motivo tienes que reconfigurar ***apache*** y/o ***PHP***, antes de comenzar de nuevo con los ./configure etc... **ELIMINA** de la caché la configuración de los ***makefiles*** viejos, esto lo puedes hacer de varias formas:

rm configure.cache o make distclean o make clean.... como quieras... y luego vuelves a repetir los pasos descritos para cada paquete con las opciones que deseas añadir.

La forma de instalar ***phplot, JPGraph y ADODB***.... esto no son librerías, son programas PHP y hay que colocarlos en el directorio apropiado para que puedan usarse, lo vemos a continuación.

Supongamos que los archivos tarball de phplot, jpgraph y adodb los hemos bajado de los links que te puse y los tenemos almacenados en /root/Taller_snort , hacemos esto:

```
cd /root/Taller_snort
cp phplot-5.0rc1.tar.bz2 /www/htdocs
cp jpgraph-1.14.tar.gz/www/htcocs
cp adodb421.tgz /www/htdocs
```

Es decir, los copiamos al directorio que aloja las webs apache.

Una vez copiados, cambiamos al directorio ***documentroot*** *de* ***apache*** *y descomprimimos allí:*

```
cd /www/htdocs
tar xjfv phplot-5.0rc1.tar.bz2
tar zvxf jpgraph-1.13.tar.gz
tar zvxf adodb421.tgz
```

Y ahora cambiamos el nombre del directorio ***jpgraph-1.14*** *como* ***jpgraph*** *y borramos los comprimidos*

```
mv jpgraph-1.13 jpgraph
rm phplot-5.0rc1.tar.bz2
rm jpgraph-1.13.tar.gz
rm adodb421.tgz
```

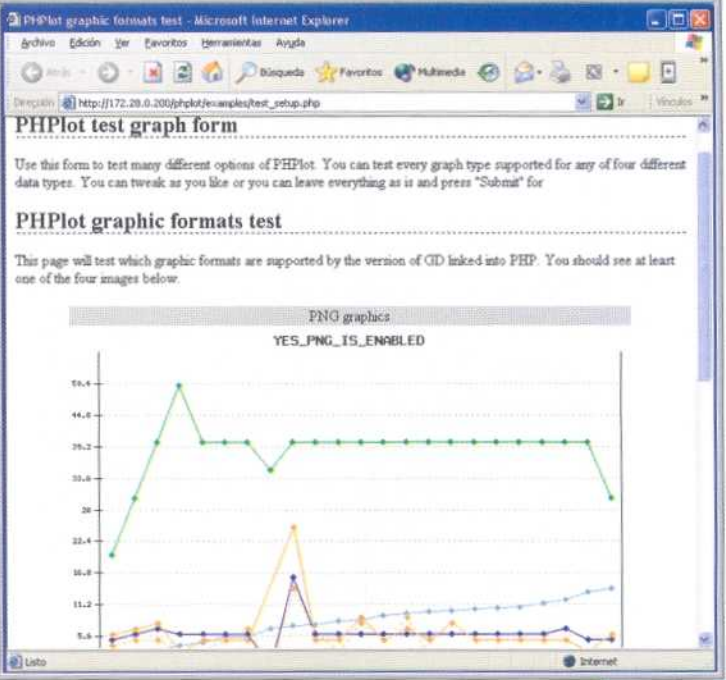
Con esto ya lo tenemos TODO preparado para usar ***ACID***

Oye!!! ¿Y ***snort****? .... ¿ Y* ***MySQL****? Pues qué va ha ser... eso ya lo tenemos...* ***Llevamos 3 meses con snort*** *y en la práctica de SAM que desarrollamos en este mismo artículo ya tenemos creada la Base de Datos y* ***snort*** *configuradito... así que....*

También podemos hacer una prueba para revisar que todo va bien... Abrimos el navegador y accedemos a esta dirección:

http://172.28.0.200/phplot/examples/test_setup.php

Y deberíamos ver una pantalla parecida a esta:



Pantalla 18.Prueba de phplot

Para probar ***jpgraph*** puedes hacerlo así: http://172.28.0.200/jpgraph/Examples/testsuit.php

Instalando ACID

Como ya he dicho, si bajaste los archivos fuente de ***snort***, en la carpeta ***./contrib*** tienes una versión de ***ACID***, pero puedes obtener las últimas revisiones en:

http://www.cert.org/kb/acid/

La última versión disponible es la ***0.9.6b23*** y puedes descargarla gratuitamente en :
http://www.andrew.cmu.edu/~rdanyliw/snort/acid-0.9.6b23.tar.gz

Antes de comenzar la instalación y configuración de ***ACID*** hay que **verificar si en el archivo snort.conf (o el archivo de configuración que elijas)** existe el ***plugin output database:*** declarado.

Como estamos *"trasteando"* es posible que no esté correcto, imaginemos que la Base de Datos que queremos usar se llama ***testids*** y que no usamos nombre de usuario ni contraseña para poder conectarnos a ella, el archivo de configuración debería incluir esta línea:

output database: log, mysql, dbname=testids host=localhost

En ***host*** puedes poner la dirección IP del servidor donde corre *MySQL* o como en el ejemplo, *localhost*, que significa el *"mismo equipo"* en donde corre ***snort***.

Además debemos de disponer de esa misma base de datos creada y con la estructura de tablas y campos que ***snort*** utiliza, para ello puedes seguir los mismos pasos que dimos para SAM, los repito sin explicarlos... con la salvedad de comentar que el ***código fuente de snort está alojado en /root/Taller_snort/Sources/snort-2.1.1*** y ya sabes que dentro de la carpeta ***contrib*** está el ***script create_mysql*** necesario para crear la estructura de la Base de Datos, bueno y ya puestos.... lo haremos de otra manera... para aprender más cosas:

```
root@linux-rh:~/Taller_snort
Archivo  Editar  Ver  Terminal  Ir  Ayuda

[root@linux-rh Taller_snort]# cp ./Sources/snort-2.1.1/contrib/create_mysql create_mysql
[root@linux-rh Taller_snort]# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11 to server version: 4.0.18-standard

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database testids;
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye
[root@linux-rh Taller_snort]# mysql -D testids<create_mysql
[root@linux-rh Taller_snort]#
```

Pantalla 19.Creación de la estructura de la Base de Datos testids

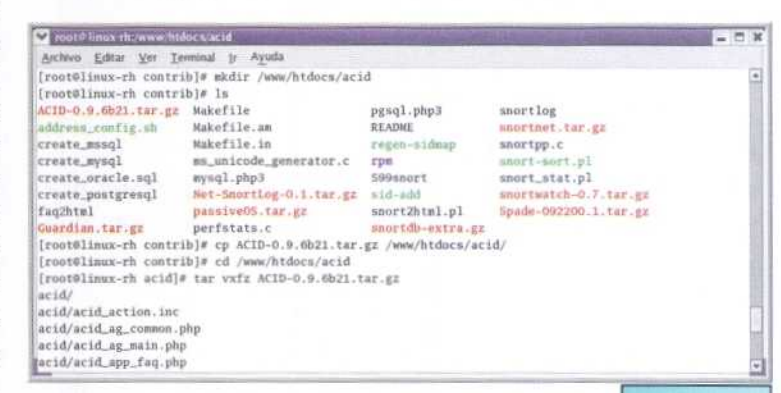
Observa que en lugar de usarse ***source create_mysql;*** tras iniciar *mysql*, simplemente creamos la base de datos, salimos del gestor y desde la línea de comandos ejecutamos la instrucción:

mysql -D testids<create_mysql la única condición para utilizar este método es que la base de datos ***testids*** estuviese creada anteriormente como hicimos con ***create database testids;***

Por cierto, verifica el lugar donde se crean las bases de datos de tu gestor *MySQL*, en el mío están en el directorio por defecto en RedHat, ***/var/lib/mysql/testids***

También puedes obtener el archivo ***create_mysql*** desde aquí (También ***ACID***) http://cvs.sourceforge.net/viewcvs.py/snort/snort/contrib/

Una vez descargado ***ACID*** (o desde la carpeta ***contrib*** de los fuentes de ***snort***) lo copiamos dentro de algún directorio de nuestro ***apache***, por ejemplo en el directorio ***acid*** que cuelga de ***/www/htdocs***



```
root@linux-rh:/www/htdocs/acid
Archivo  Editar  Ver  Terminal  Ir  Ayuda
[root@linux-rh contrib]# mkdir /www/htdocs/acid
[root@linux-rh contrib]# ls
ACID-0.9.6b21.tar.gz  Makefile                  pgsql.php3        snortlog
address_config.sh     Makefile.am               README            snortnet.tar.gz
create_mssql          Makefile.in               regen-sidmap      snortpp.c
create_mysql          ms_unicode_generator.c    rpm               snort-sort.pl
create_oracle.sql     mysql.php3                S99snort          snort_stat.pl
create_postgresql     Net-SnortLog-0.1.tar.gz   sid-add           snortwatch-0.7.tar.gz
faq2html              passiveOS.tar.gz          snort2html.pl     Spade-092200.1.tar.gz
Guardian.tar.gz       perfstats.c               snortdb-extra.gz
[root@linux-rh contrib]# cp ACID-0.9.6b21.tar.gz /www/htdocs/acid/
[root@linux-rh contrib]# cd /www/htdocs/acid
[root@linux-rh acid]# tar vxfz ACID-0.9.6b21.tar.gz
acid/
acid/acid_action.inc
acid/acid_ag_common.php
acid/acid_ag_main.php
acid/acid_app_faq.php
```

Pantalla 20. descomprimir ACID en /www/htdocs/acid

Por si no lo ves claro en la pantalla, la secuencia de comandos es:

mkdir /www/htdocs/acid
ls
cp ACID-0.9.6b21.tar.gz /www/htdocs/acid
cd /www/htdocs/acid
tar vxfz ACID-0.96b21.tar.gz

Realmente ***se pueden mantener varias copias de ACID*** *e incluso enlazadas a diferentes bases de datos, simplemente te*

*vas creando otros directorios diferentes dentro de **htdocs** y les pones otros nombres, así puedes generar diferentes informes, **cada copia funcionará de forma independiente.***

Dentro del directorio que nos creamos con ***mkdir acid***, existirá "*otro*" llamado igual, *acid*... es decir ***la ruta correcta será /www/htdocs/acid/acid/....***

Y dentro de ese último directorio *acid*, sólo existen archivos... el que nos interesa es el **archivo de configuración de acid, que se llama: *acid_conf.php*** Accedemos a él:

gedit /www/htdocs/acid/acid/acid_conf.php

Hay que modificar algunas líneas, veamos una a una con los valores por omisión que tienen:

$Dblib_path = ""; Aquí tendremos que poner la ***ubicación exacta de adodbc***,

$Dblib_path = "/www/htdocs/adodb";

$Dbtype= "mysql"; es decir el **gestor** que queremos usar, se queda como está.

***$alert_dbname = "snort_log";* nombre de la base de datos** de alertas, hay que poner:

$alert_dbaname = "testids";

$alert_host = "localhost"; que será el *host* donde corre el **servidor de MySQL,**

$alert_host = "172.28.0.200";

$alert_port = ""; que es el **puerto** por donde escucha **MySQL**, escribimos:

$alert_port = "3306";

$alert_user = "root"; el **usuario con privilegios suficientes para acceder a la Base de datos**, como no incluimos usuarios ni contraseñas, vamos a tener un problema... pero vamos a ponerlo y ya veremos:

$alert_user = "usuacid";
$alert_password = "mitesoro";

Otras variables muy parecidas a las que acabamos de describir son:

$archive_dbname = "snort_archive";
$archive_host = "localhost";
$archive_port = "";
$archive_user = "root";
$archive_password = "mypassword";

Que permitirán **mantener una copia de seguridad de las alertas** con los parámetros que se especifiquen, por ejemplo ponemos:

$archive_dbname = "testids_copia";
$archive_host = "172.28.0.200";
$archive_port = "3306";
$archive_user = "usuacid";
$archive_password = "mitesoro";

$ChartLib_path =""; aquí debemos poner la ***ruta exacta de las librerías phplot***, es decir

$ChartLib_path ="/www/htdocs/phplot";

***$chart_file_format = "png";* formato de ficheros gráficos**, pueden ser gif, jpeg o png, como nosotros instalamos *libpng* **lo dejamos como está.**

Luego hay otras variables que permiten configurar el número de alertas, las filas y columnas para los resultados, el color de fondo, la rejilla, etc... estas las dejaremos como están y pasamos a:

$portscan_file = ""; en el cual podríamos incluir el ***fichero de logs que crean los preprocesadores portscan y portscan2***, normalmente se deja en blanco, pero si quieres verlo con el *plugin* aquí se indicaría la ruta hacia el archivo que generan los *preprocesadores* indicados.



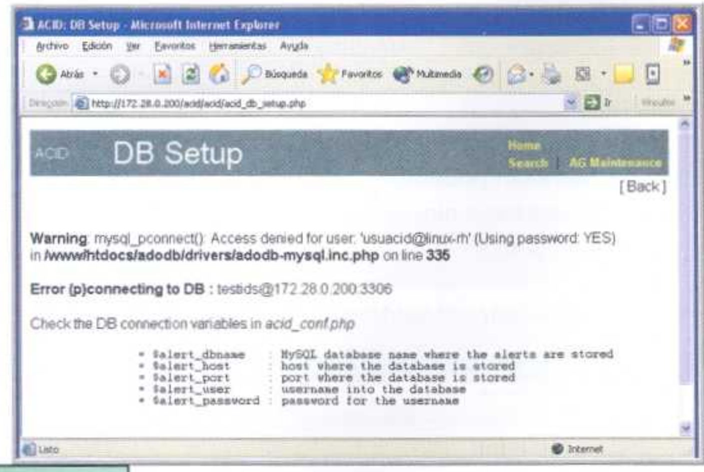
IMPORTANTE

Estamos cometiendo un error GRAVE, ni usamos usuarios con contraseñas para las Bases de Datos, ni estamos

Asegurando el WebServer Apache, vamos que cualquiera podrá conectarse a la base de datos o al servidor y ver las alertas... y hasta eliminarlas... vamos "para nota",

También hay otras variables para especificar el mail al cual se deben enviar las alertas, bueno esas son muy sencillitas y ahora no son necesarias que las indiquemos para que funcione...

Guardamos los cambios en el archivo de configuración de ***ACID***.... y navegamos hacia la página:
http://172.28.0.200/acid/acid/acid_db_setup.php



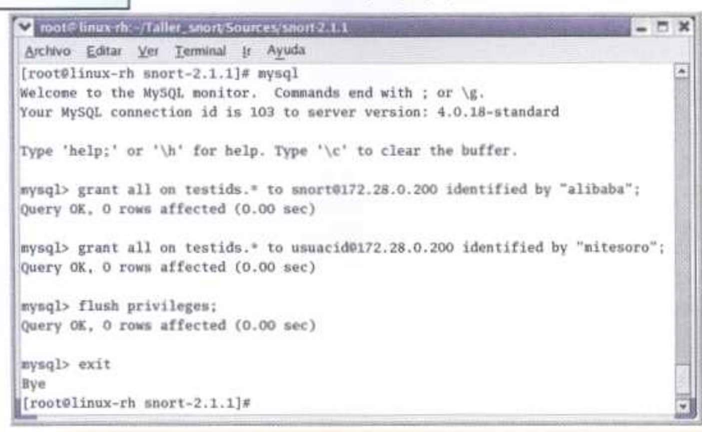
Pantalla 21.Setup de ACID con errores de autenticación

Y lo que nos esperábamos... ***un error de autenticación***

Así que no nos queda otro remedio que crear el ***usuario usuacid*** con la ***contraseña mitesoro*** en *mysql* para que podamos seguir.... y ya puestos... también crearemos usuario y *password* para snort, que serán ***usuario:snort, contraseña: alibaba***

Arrancamos *mysql* y ponemos:

Pantalla 22.Creación de usuarios y Contraseñas



Luego **editamos el archivo** ***/etc/snort/snort.conf*** y cambiamos:

output database: log, mysql, dbname=testids host=localhost user=snort password=alibaba
(todo en una línea)

Y guardamos el archivo de configuración.



Realmente...

Realmente cuando creamos los dos usuarios no hubiese sido preciso otorgar todos (*all*) los privilegios sobre las tablas de *testids* (***testids.****) , nos bastaría con:

mysql

grant INSERT, UPDATE, SELECT, CREATE, DELETE on testids. to usuacid@172.28.0.200 identified by "mitesoro"*

grant INSERT, UPDATE, SELECT, CREATE, DELETE on testids. to snort@172.28.0.200 identified by "alibaba"*

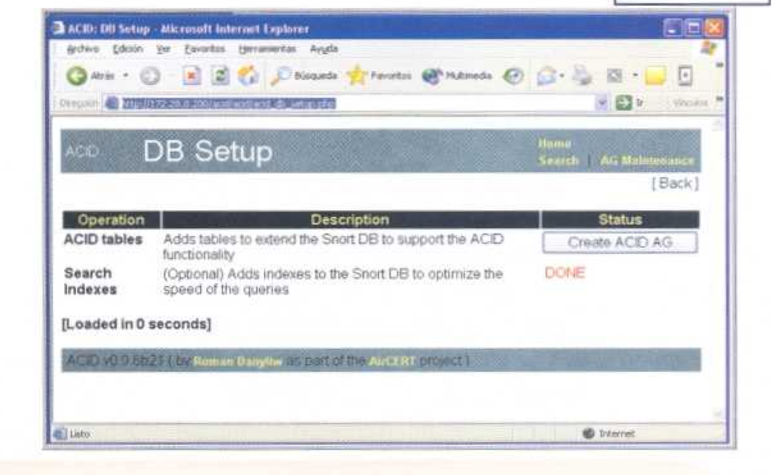
flush privileges;

exit

El caso es que probamos de nuevo con la dirección

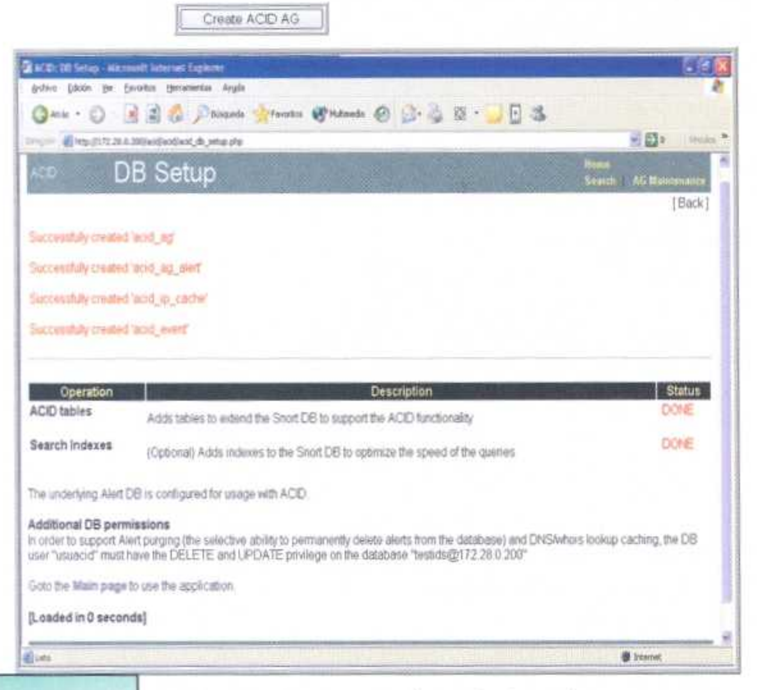
http://172.28.0.200/acid/acid/acid_db_setup.php Y......

Pantalla 23.Setup ACID correcto



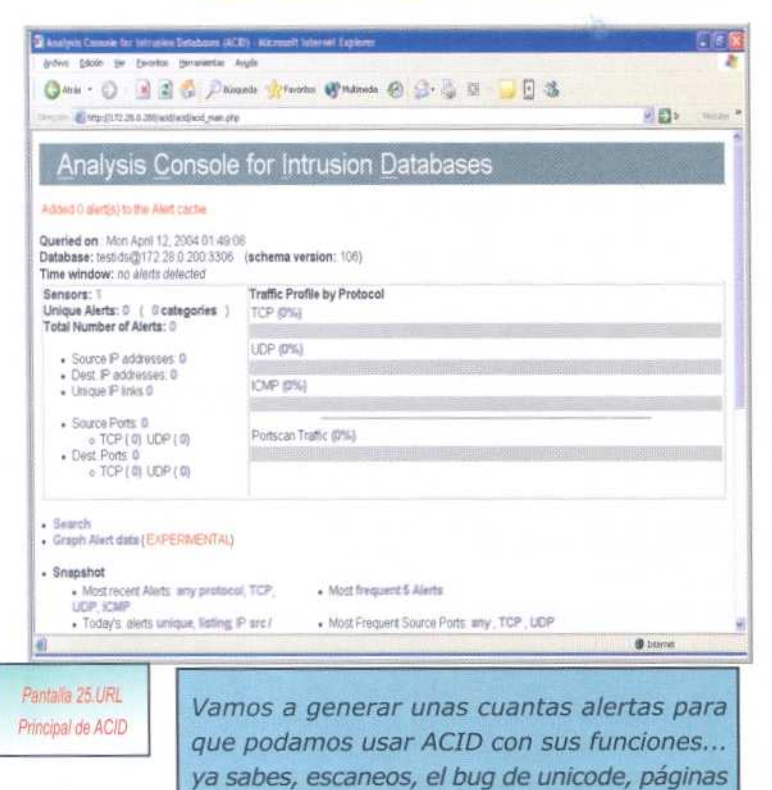
Y pulsamos en el botón ***Create ACID AG***

Create ACID AG



Pantalla 24. Fin de proceso de configuración de ACID

Ahora Pulsamos arriba a la derecha en ***Home*** y/o navegamos a la URL que tendremos que acceder a partir de ahora: http://172.28.0.200/acid/acid/acid_main.php



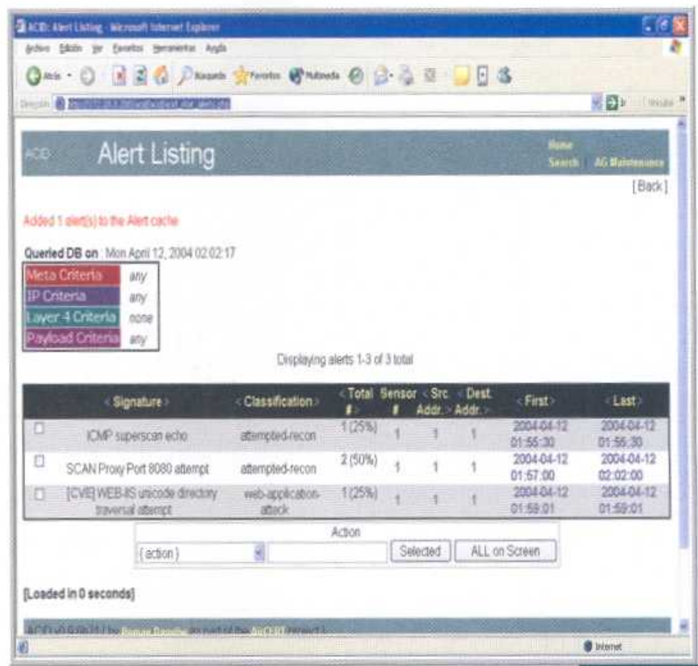
Pantalla 25.URL Principal de ACID

Vamos a generar unas cuantas alertas para que podamos usar ACID con sus funciones... ya sabes, escaneos, el bug de unicode, páginas "prohibidas", lo que se te ocurra...

Pantalla 26 Alertas de ACID

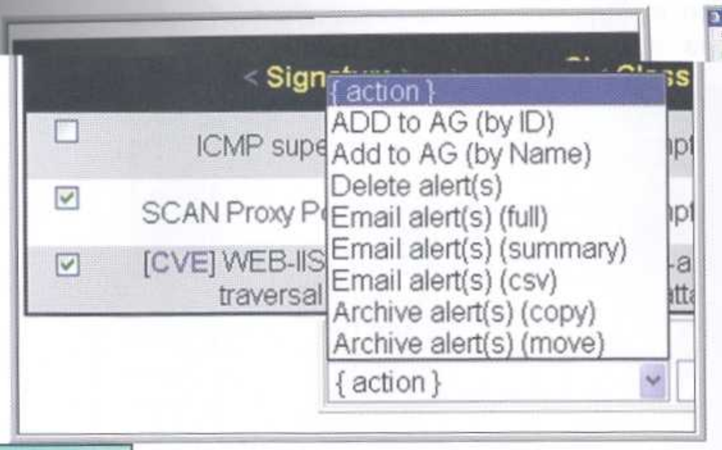
Me limité a un escaneo y a tentar el bug de unicode, también podemos usar esta url para ver los resultados o pinchar en los links de la página principal....

http://172.28.0.200/acid/acid/acid_stat_alerts.php



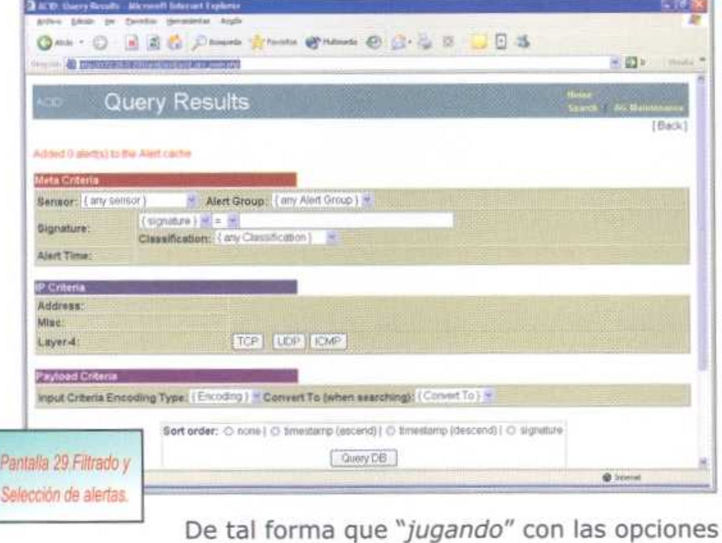
Pantalla 27 Listado de alertas

Podemos seleccionar las alertas verificando las casillas correspondientes y pulsando en el desplegable que pone ***{action}*** enviarlas...



Pantalla 28 Filtrado y Selección de alertas

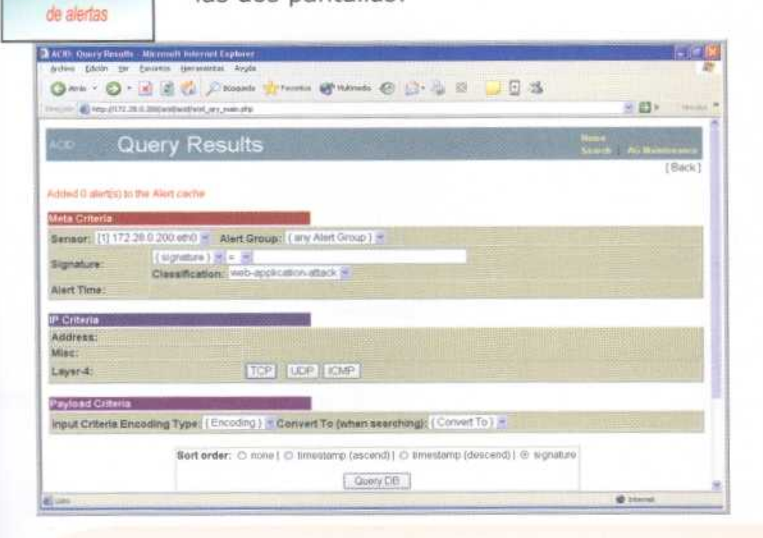
O filtrarlas.... navegando a la dirección: **http://172.28.0.200/acid/acid/acid_qry_main.php**

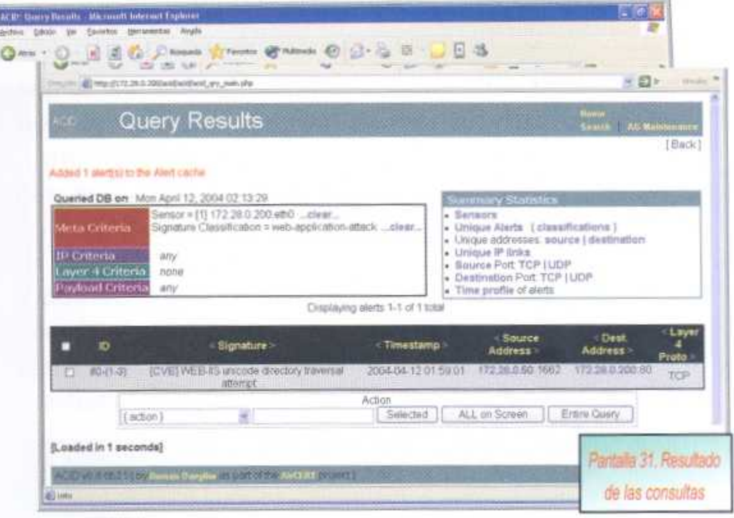


Pantalla 29 Filtrado y Selección de alertas.

De tal forma que *"jugando"* con las opciones disponibles podemos filtrar la información y por último pulsamos en ***Query DB***... te pongo las dos pantallas:

Pantalla 30 Consulta de alertas

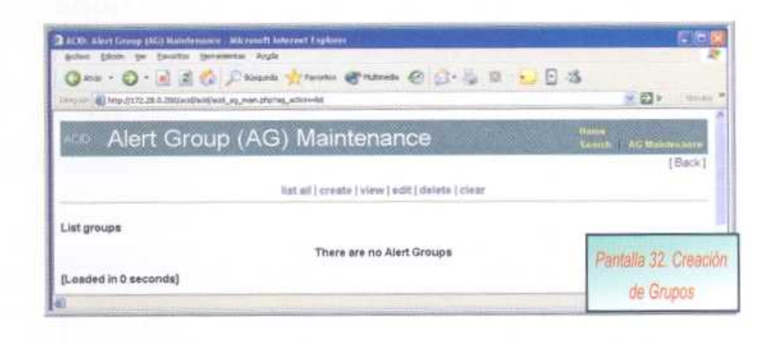




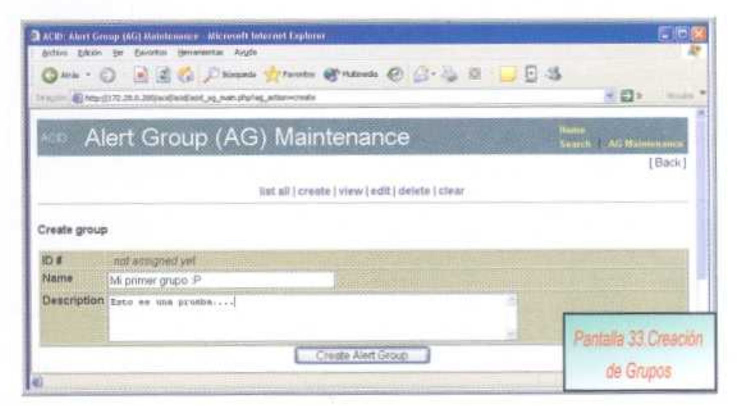
Pantalla 31 Resultado de las consultas

Si pinchamos en ***AG Maintenance*** (arriba a la derecha) podemos ***crear grupos de alertas y mantener los mismos...*** también se puede hacer eso mismo navegado a la URL

http://172.28.0.200/acid/acid/acid_ag_main.php?ag_action=list

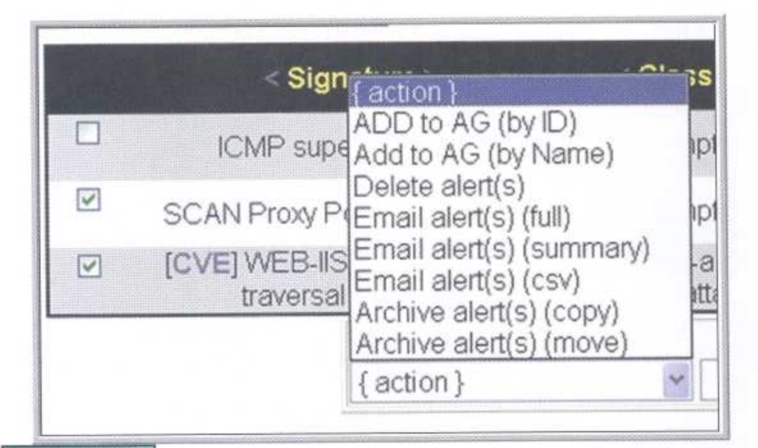


Pantalla 32 Creación de Grupos



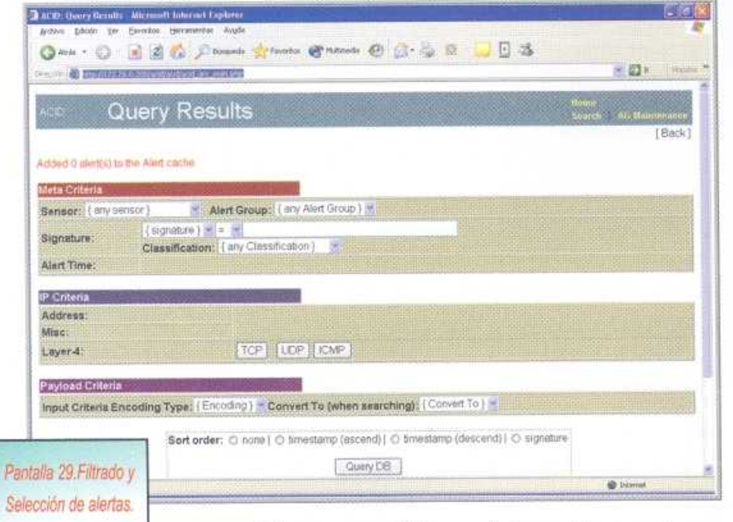
Pantalla 33 Creación de Grupos

El **uso de grupos nos permite mantener juntas determinadas alertas**, por ejemplo si estamos interesados en tener juntitas todas



Pantalla 28.Filtrado y Selección de alertas

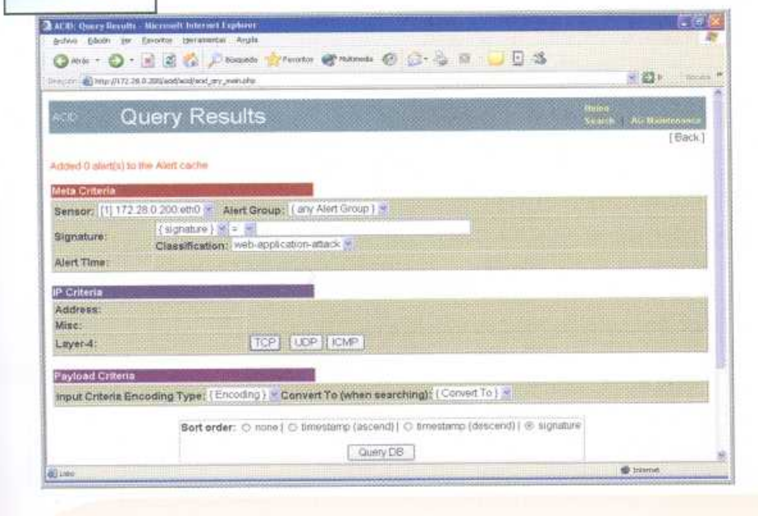
O filtrarlas.... navegando a la dirección:
http://172.28.0.200/acid/acid/acid_qry_main.php

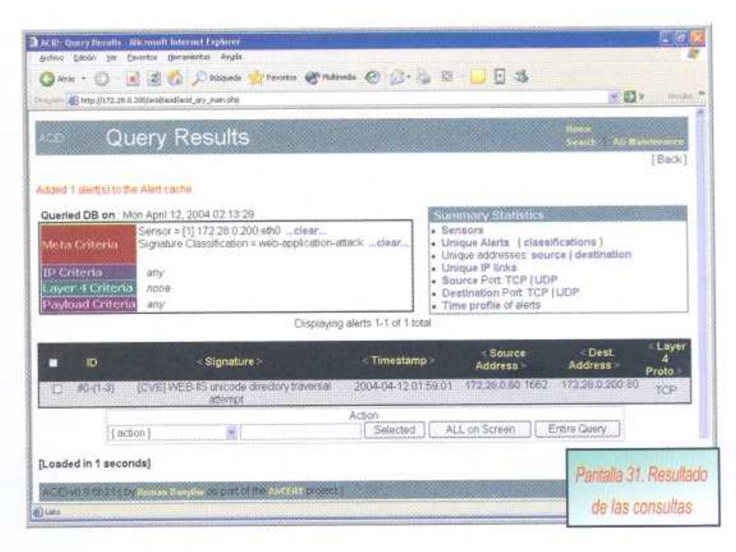


Pantalla 29.Filtrado y Selección de alertas.

De tal forma que *"jugando"* con las opciones disponibles podemos filtrar la información y por último pulsamos en ***Query DB***... te pongo las dos pantallas:

Pantalla 30.Consulta de alertas

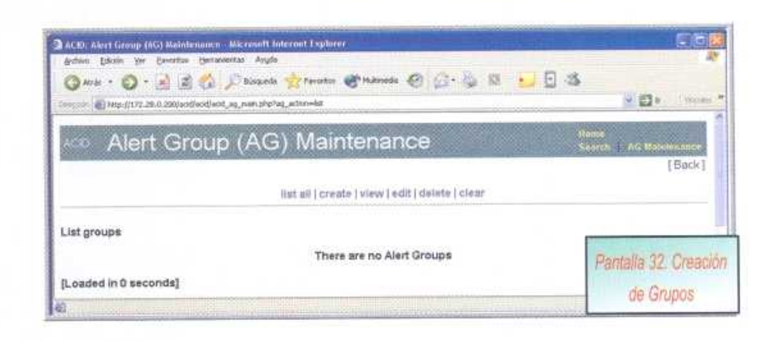




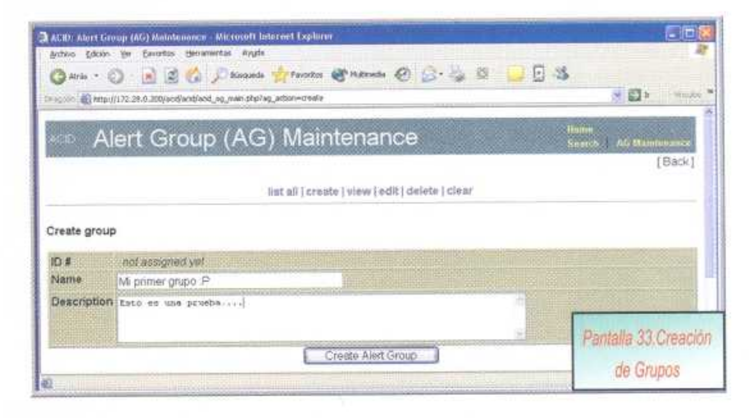
Pantalla 31. Resultado de las consultas

Si pinchamos en ***AG Maintenance*** (arriba a la derecha) podemos ***crear grupos de alertas y mantener los mismos...*** también se puede hacer eso mismo navegado a la URL

http://172.28.0.200/acid/acid/acid_ag_main.php?ag_action=list



Pantalla 32. Creación de Grupos



Pantalla 33.Creación de Grupos

El **uso de grupos nos permite mantener juntas determinadas alertas**, por ejemplo si estamos interesados en tener juntitas todas

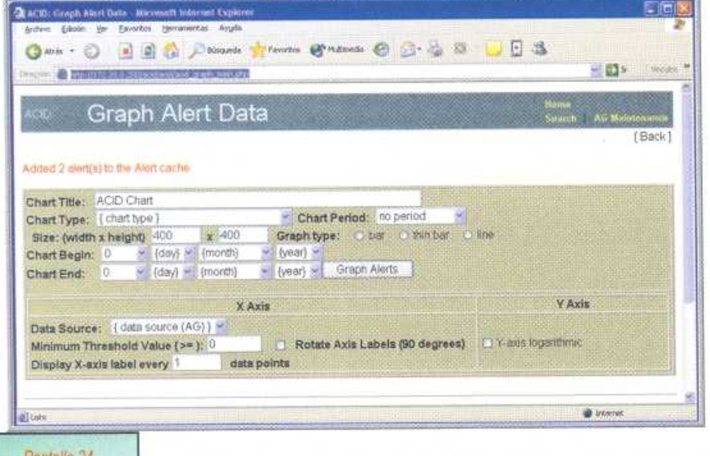
las alarmas y registros de determinado tipo de intrusión , en fin, no deja de ser una "*aplicación web*" sencillita en el uso... lo que hace fuerte a ***ACID*** es su ***integración con snort, con Apache, con MySQL*** y claro... un mantenimiento apropiado.... pero eso dependerá de ti.

Hay que "*practicar*" para sacarle partido y probar... lo importante es lo que ya hemos aprendido, configurar ***snort***.... ***ACID*** no deja de ser una forma más de ver los resultados...

Ahh!! Y por supuesto... aunque en las pantallas que estoy poniendo de ***ACID*** verás que el navegador es Internet Explorer, es decir, lo "*veo*" desde un *Windows*... tanto ***ACID***, como ***snort***, como ***Apache***, como ***MySQL*** *están corriendo en un LiNUX RedHat*.... el navegador es una "*herramienta*" no el medio...

Uyy!!! Casi se me olvida... los gráficos.... que para eso los instalamos...

Accedemos a la dirección:
http://172.28.0.200/acid/acid/acid_graph_main.php



Pantalla 34. Generación de Gráficos

Y tras seleccionar los periodos correctos veremos el gráfico... pero te recuerdo que lo principal está aquí:

http://172.28.0.200/acid/acid/acid_main.php

Y a través de ese link podrás ir "*navegando*" por ***ACID***, no todos los links aparecen subrayados, unos están en amarillo, otros en azul... pero bueno... eso es "*Navegar*"...

Si quieres asegurar "*un poquito más*" el acceso a las páginas de ***ACID***, puedes hacer lo que sigue:

```
cd /www/bin
./htpasswd -c /www/bin/password admin
```

*(y escribes una contraseña, por ejemplo)***:**
acid4mi

Con esto acabamos de asociar la ***contraseña acid4mi al usuario admin.***

Ahora hay que **modificar el archivo de configuración de apache**

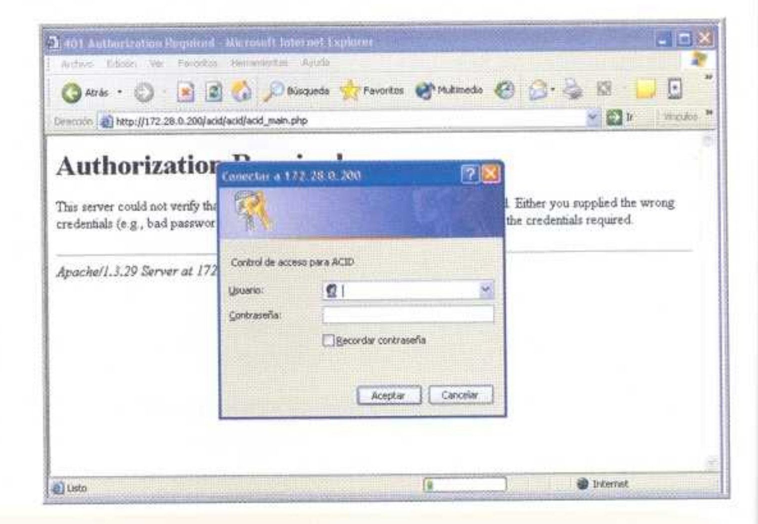
```
gedit /www/conf/httpd.conf
```

Y añadimos las siguientes líneas en la sección correspondiente:

```
<Directory "/www/htdocs/acid/acid">
AuthType Basic
AuthName "Control de acceso para ACID"
AuthUserFile /www/bin/password
Require user admin
AllowOverride None
</Directory>
```

Después reinicia el servidor web
/www/bin/httpd restart

Ahora cuando se intente acceder a cualquier página dentro del directorio de acid, nos pedirá la contraseña para el usuario admin. Antes de mostrar dicha página, el pass era acid4mi



Bien.... y nos queda *Windows*.... mmm, para , ***te propongo una cosa:***

¿Qué te parece si te cuento una cosa NUEVA para Windows y la configuración de ACID+snort+MySQL+PHP+Apache (o IIS) y te lo cuelgo en un link? .... porque repetir el "rollo" de otra forma... como que no me apetece y llenamos la revista innecesariamente.

Creo que es la mejor opción. Encontrarás una guía minuciosa y detallada para hacer esto mismo pero en plataformas Windows en la sección Portadas y Descargas de la Web de la revista (www.hackxcrack.com) y también en este enlace (por si acaso la Web no está actualizada):

http://www.forohxc.com/snort/mswin/snort.plugins.win.pdf

Y ahora veamos una herramienta específica de ***Windows***, ***MUY BUENA y GRATIS***.... se llama: IDS Center para Windows.

IDS Center para Windows

Una de las formas más sencillas de configurar **snort** para *Windows* es usar ***IDSCenter***.

No se trata de un *plug-in,* es un entorno gráfico para configurar snort de forma centralizada desde una misma aplicación, *un front-end*, lo puedes descargar de:

http://www.engagesecurity.com/downloads/idscenter/idscenter11rc4.zip

Y otras herramientas también en: http://www.engagesecurity.com/downloads/#idscenter

Sus principales características son:

Soporte para versiones de ***snort*** 2.0, 1.9 y 1.8

Control de ***snort*** como servicio

Configuración gráfica de variables, *preprocesador*, *plug-ins* de salida y *reglas*

Editor de reglas

Plug-ins especiales para interactuar con *Firewalls* como BlackIce y otros

Notificación de alertas y alarmas sonoras, mail, etc.

Rotación de logs y Acceso a Bases de datos (*MySQL, Postgre, Oracle, SQL server...*

Visores de texto, *HTML, XML*

Posibilidad de ejecutar programas determinados cuando una alerta es detectada

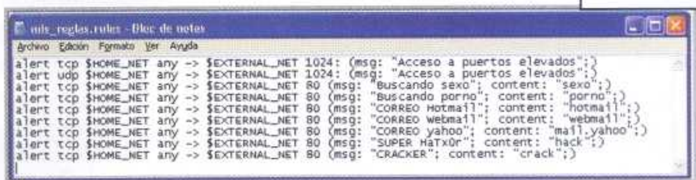
Actualizaciones *on-line* de reglas, ***snort inline***,

Incluir las alertas como registros dentro del visor de sucesos de *Windows*

Soporte para otros *plug-ins* como *ACID, SnortSnarf y Webmin*

Antes de instalar *IDScenter*, vamos a retomar uno de los ejercicios que hicimos anteriormente, para ello nos creamos un **fichero de reglas** con este contenido y lo guardas en:

C:\snort\rules\mis_reglas.rules



```
alert tcp $HOME_NET any -> $EXTERNAL_NET 1024: (msg: "Acceso a puertos elevados";)
alert udp $HOME_NET any -> $EXTERNAL_NET 1024: (msg: "Acceso a puertos elevados";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "Buscando sexo"; content: "sexo";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "Buscando porno"; content: "porno";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "CORREO Hotmail"; content: "hotmail";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "CORREO webmail"; content: "webmail";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "CORREO yahoo"; content: "mail.yahoo";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "SUPER HaTxOr"; content: "hack";)
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "CRACKER"; content: "crack";)
```

Pantalla 35.Nuevo archivo de reglas. Mis_reglas.rules

Ahora vamos a crear el **archivo de configuración** y lo guardamos en : ***C:\snort\etc\mi_snort.conf***



```
var HOME_NET 172.28.0.0/16
var HTTP_PORTS 80
var EXTERNAL_NET any
var RULE_PATH C:\snort\rules
output database: log, mysql, dbname=idstest host=localhost
include $RULE_PATH\mis_reglas.rules
```

Pantalla 36.Nuevo archivo de configuración. Mi_snort.conf

Observa que hemos incluido ***output database:*** es decir, necesitaremos que *MySQL* esté corriendo y acceso a esa Base de Datos... pero no deberías tener problemas, porque es la misma que usamos con ***SAM***.

También recuerda cambiar el contenido de la variable ***HOME_NET*** por el rango de red/subrerd que usas.

Ahora Instalemos ***IDS Center*** para *Windows, la instalación.... mejor ni la comento, las pautas de siempre... ejecutar el setup del programa y Siguiente-siguiente-siguiente...*

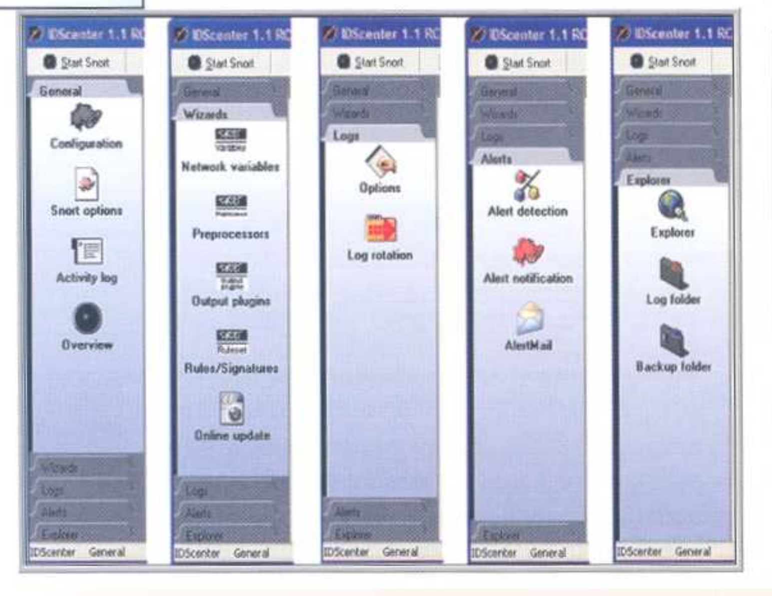
En el escritorio aparecerá un acceso directo a ***IDScenter*** y cuando lo ejecutes aparecerá otro icono más en la bandeja del sistema (*ese negro con la banda roja*)

Al ***pulsar el botón derecho*** del ratón sobre es icono, verás el ***menú contextual de IDS Center*** como muestra la pantalla.

Pantalla 37.Configuración de IDSCenter

Para configurar ***IDS Cent***er hay que ***seleccionar Settings...***

El uso de ***IDS Center*** es muy simple, en la **zona izquierda tenemos una serie de fichas** que incluyen diferentes opciones para configurar a ***snort***, a continuación te pongo una pantalla con las opciones disponibles:



Pantalla 38.Opciones disponibles para configurar IDSCenter

Basta con ir pinchando en cada una de ellas y se irán desplegando sus contenidos.

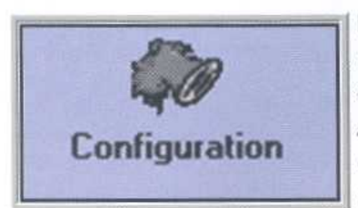
En la ***zona superior de IDS Center*** Tenemos las herramientas para iniciar ***snort*** (o detenerlo cuando está en marcha), el visor de alertas, *Resetear* las alarmas, ***test*** de configuración, recargar el sistema y Aplicar (***Apply***) los cambios efectuados.

Pantalla 39.Detalle de la zona superior. IDSCenter



Empezaremos por General que además será la primera pantalla que apareció cuando pulsamos en ***settings*** desde el icono de la bandeja del sistema.

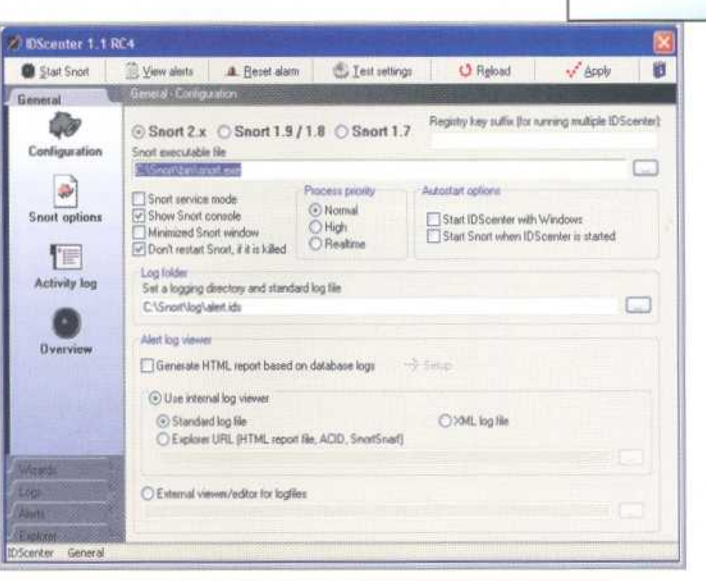
Disponemos **de 4 opciones** dentro del mismo.



Para indicar el la ruta al ejecutable de ***snort***, si ha de correr como servicio o en modo consola, la prioridad del proceso (Normal, alta o tiempo real) Si deseamos iniciar ***snort*** junto con *Windows* al inicio, el camino hacia el ***archivo alert.ids***, formatos para los visores de logs (*HTML, XML, Estándar*).

Usaremos la misma configuración que se muestra en esta pantalla:

Pantalla 40. Opción Configuration de IDSCenter





Ahora seleccionaremos ***Snort Options***, lo más importante de esta opción es **indicar la ubicación del archivo de configuración**, además en la zona central verás el contenido del mismo *"modificado"* por ***IDSCenter***, le pone una serie de comentarios ***(#)*** y lo organiza por secciones.

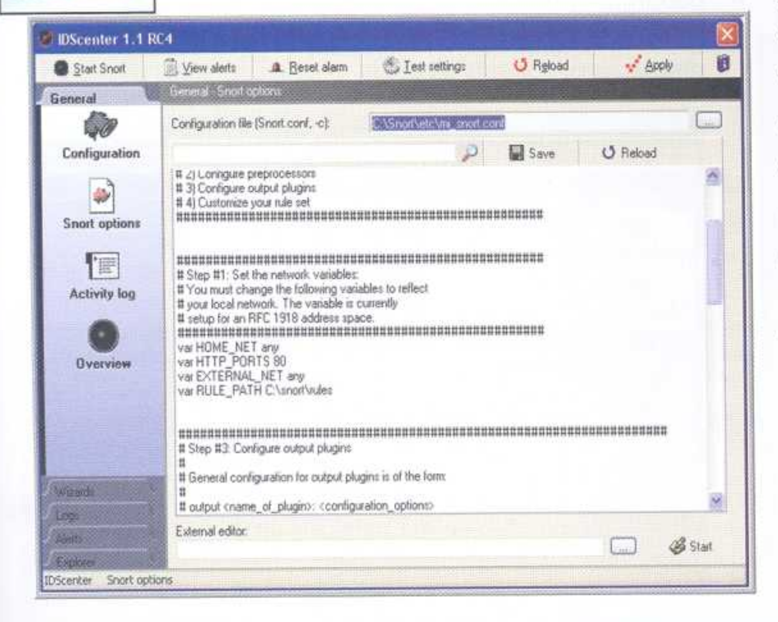
Puedes ***navegar por la estructura de directorios*** del disco duro si pinchas en [ ] esto valdrá para otras muchas opciones.

También podemos modificarlo directamente e ir configurando allí ***snort***... pero espera no lo hagas... también podemos usar la barra de herramientas que hay embebida en la pantalla para ***buscar información*** (la lupa) ***recargar el archivo o salvar los cambios efectuados***.



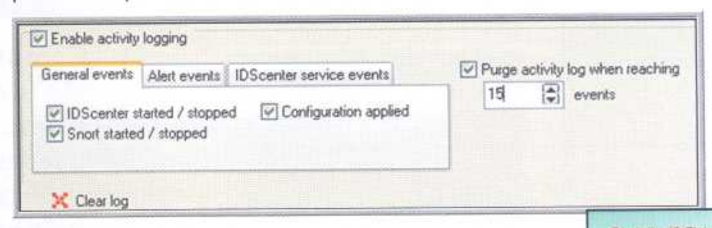
En nuestro caso lo que tenemos que hacer es indicar la ruta correcta y el nombre del archivo de configuración que como ya lo tenemos creado se llama ***mi_snort.conf***

Pantalla 41. Snort options de IDSCenter





Ahora seguimos bajando... y pulsamos en ***Activity Log***. Aquí podemos monitorizar mediante sucesos la **propia actividad de *IDSCenter***, estos logs **no tienen nada que ver con *snort***, sólo informan de las veces que se ha iniciado ***IDSCenter***, las horas, las paradas, los accesos a las bases de datos, etc.. **La pantalla central tiene tres fichas** de configuración, puedes probar con ellas, son fáciles de entender y para esta práctica no es relevante su contenido.



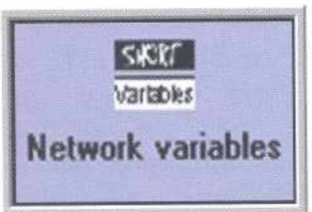
Pantalla 42.Fichas disponibles en Activity Log



La última opción es ***Overview***.. que ahora no contiene nada... y mejor que así sea siempre, porque en esta pantalla se mostrarían los **errores de configuración** si los hemos cometido y también a modo de indicación, nos muestra la orden que deberíamos teclear en la línea de comandos en el caso de usar la *shell* para arrancar ***snort***.

Vamos por la segunda ficha de menús... ***Ficha de Wizards***

Esta es una de las más importantes, por no decir la fundamental, puesto que a través de la misma daremos plena funcionabilidad a ***snort***, prácticamente todo lo que hemos hecho durante estos tres meses con ***snort***, se desarrolla aquí.



Lo primero que nos encontramos es con esta opción, en ella podemos **definir las**

variables que usará *snort*, incluso podemos modificarlas, borrarlas, añadir otras nuevas, vamos es como si fuese un editor de variables. Como ya tenemos alguna que otra creada, nos aparecerán en la zona central, como muestra el ejemplo.

Pantalla 43. Variables de snort por IDSCenter

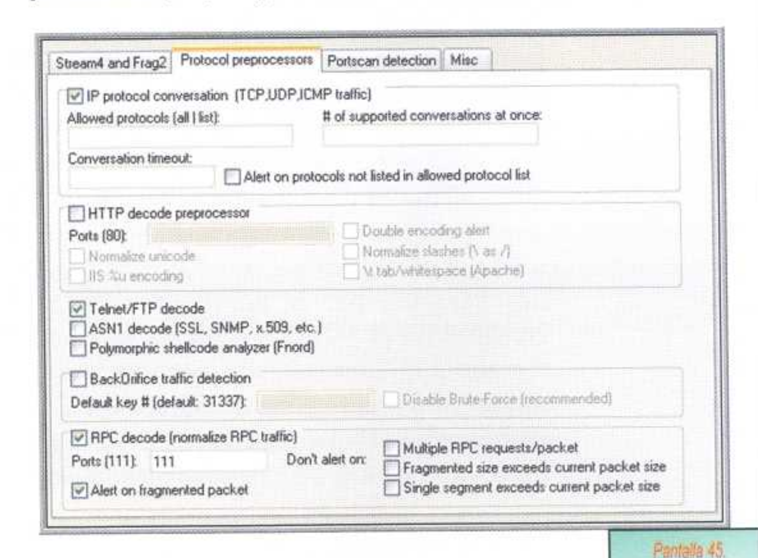


Pulsamos más abajo... en ***Preprocessors***... Como te puedes imaginar en esta ficha podemos ***configurar las directivas del preprocesador*** que queramos aplicar, *te remito al artículo del mes pasado donde se explicaban con detalle cada una de ellas, en la parte central de la pantalla, tenemos asu vez fichas para configurar los preprocesadores **stream4**, **frag2**, **protocolos y miscelánea**.*

Pantalla 44. Directivas para preprocesadores stream4 y frag2 desde IDSCenter

Directivas para los ***preprocesadores Stream4 y Frag2***

Directivas para el preprocesador de protocolos, IP, http, Telnet, FTP, RPC, fnord...



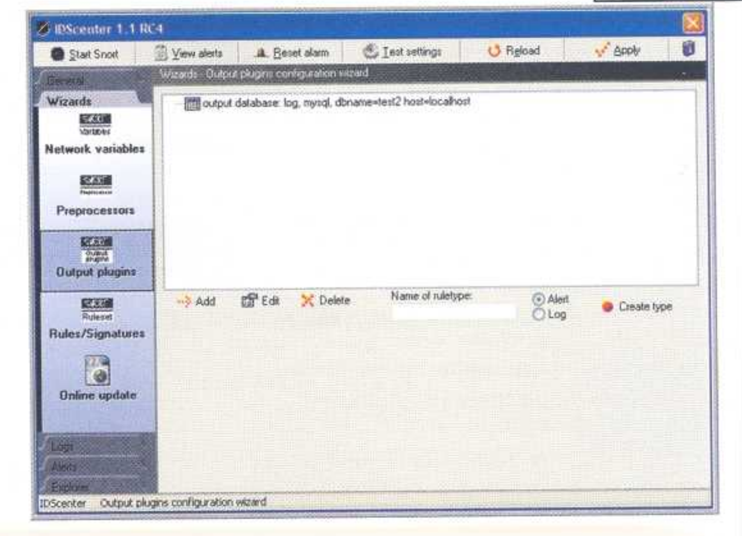
Pantalla 45. Directivas para preprocesadores de protocolos desde IDSCenter

Las fichas de *PortScan detection* (***preprocesadores portscan y portscan2***) y la de ***Misc*** (***preprocesador ARP spoofing***) aunque muy interesantes no son relevantes para este ejercicio.



Continuamos bajando... y pinchamos en ***Output Plugins***, aquí aparecerán los ***plugins de salida*** que tengamos construidos, también podemos generar otros nuevos o modificar los existentes con las herramientas ***Add, Edit, delete., etc*** que nos ofrece la misma pantalla.


Pantalla 46. Plugins de salida desde IDSCenter



Te recomiendo que tras este ejercicio regreses a esta pantalla y pruebes un poquito con las opciones disponibles, es increíblemente fácil de usar y bastante completo, puedes seleccionar bases de datos *MySQL, ORACLE, Postgre*, puedes especificar los puertos de escucha de los servidores, los usuarios y contraseñas... mucho más fácil que aprenderse las sintaxis y escribirlas en el archivo de configuración, verdad?

Ruleset

Rules/Signatures

Seguimos dentro de ***Wizard***, pero le toca a ***Rules/ Signatures.***

Te lo imaginas, lo sé... sé que eres aplicado.... Efectivamente esto es para "*manejar*" el ***archivo de reglas*** y/o incluir las reglas directamente.

Entre las cosas más importantes que tenemos que hacer aquí es la de indicar cual va a ser nuestro archivo de reglas y dónde debe ir a buscarlo ***snort***...

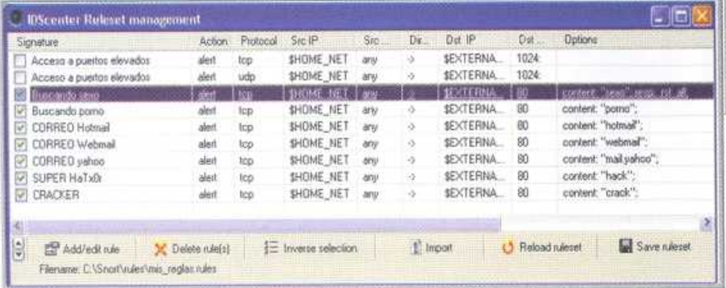


Pantalla 47. Ubicación del archivo de configuración de snort desde IDSCenter

Incluso podemos incluir más de uno... pero no queda sólo ahí... ***esto es mucho más POTENTE!!!***

Vamos ha hacer ***doble-clic sobre la regla que se muestra verificada***, pero no en la casilla de verificación... **SOBRE el nombre**... y verás esto:

Pantalla 48. Seleccionar y modificar reglas desde IDSCenter



Aquí puedes modificar la opción ***action, protocolo, red origen, puerto origen, dirección, red destino, puerto destino y opciones de contenido y flujo***, vamos lo mismo que hacíamos en el mes anterior con el bloc de notas pero ***con interface gráfico***... sólo hace falta pulsar, hacer ***doble clic***, en la regla que queramos modificar o quitar la verificación de las casillas y no se tomará en cuenta la regla, pero no se borra.

También podemos incluir, borrar, insertar, importar nuevas reglas...

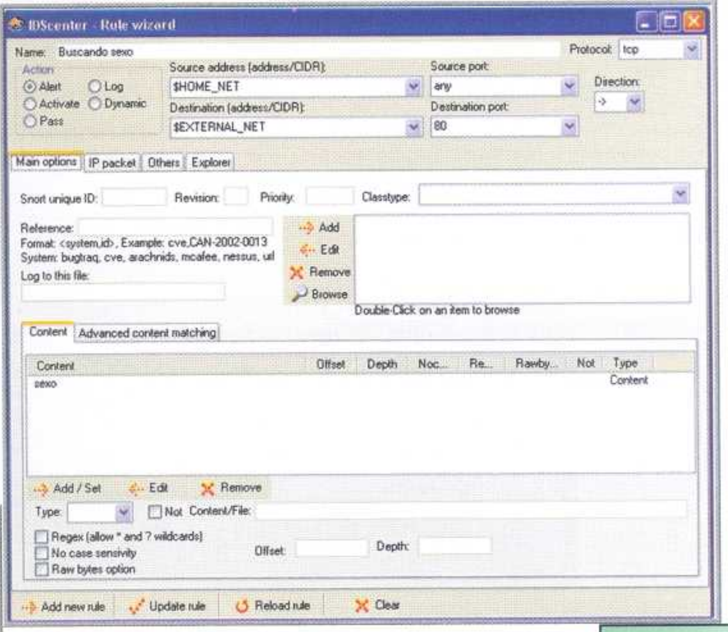
Pantalla 49. Detalle del manejo de reglas desde IDSCenter



PERO HAY MAS.... ALGO MUY UTIL!!!

Pongamos que deseamos modificar y personalizar una de ella, por ejemplo la de *sexo*...

Así que pulsamos dos veces sobre el nombre *Buscando sexo* (no sobre la casilla de verificación, sobre la regla en sí misma)



Tenemos control TOTAL de todas las opciones para esa regla en particular, podemos cambiar sus parámetros y con ello el comportamiento de ***snort***... observa que dentro

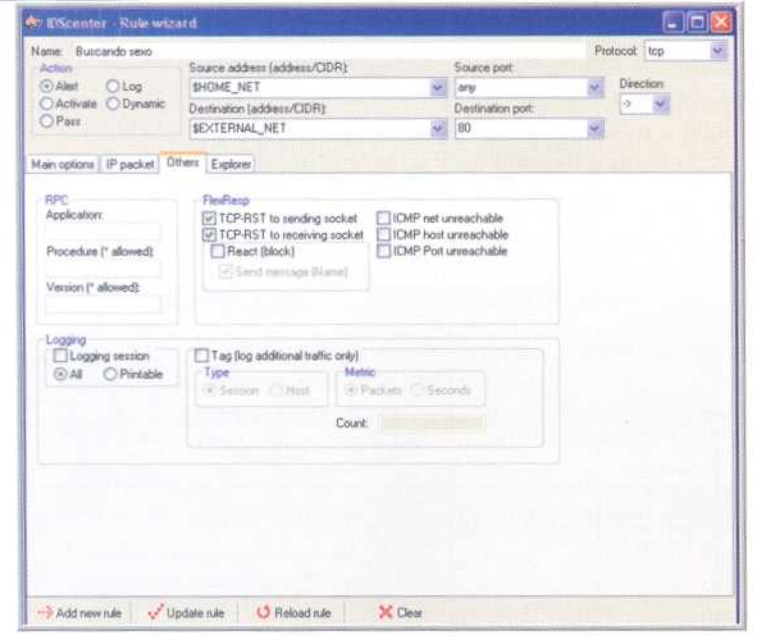
Pantalla 50. Control total del contenido de reglas desde IDSCenter

de la misma pantalla hay varias fichas, es muy, muy potente y fácil de configurar si sabes de lo que va cada cosa... y ***te recuerdo OTRA VEZ*** que lo que significa cada una de esas opciones las vimos el mes pasado con todo lujo de detalles.

Por ejemplo, algo que ya sabemos y que hemos usado en otros ejercicios....

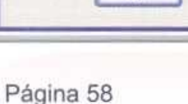
Tal y como está definida la regla, sólo alertará de que alguien accede a páginas de sexo pero no *"cortará"* la comunicación, si deseamos que lo haga, pincha en la ficha que pone ***Others*** y en el apartado ***FlexResp*** verifica si *"matamos"* la comunicación desde el que lo envía o desde el que lo recibe... como se muestra en la siguiente pantalla:

Pantalla 51. Modificación de la regla desde IDSCenter



*Buff. No puedo extenderme en cada opción... para eso llevamos muchos meses con **snort**, cada una de esas casillas, opciones, desplegables, etc... Los hemos comentado con bastante profundidad, si tienes a mano los otros dos artículos está TODO dicho.*





Una vez hechas las modificaciones pertinentes, pincha ***en Update rule*** (en la zona inferior de la pantalla) y te saldrá un cartelito advirtiendo de los cambios efectuados:

Luego ves cerrando las ventanas que se abrieron y te volverá de nuevo a preguntar si deseas ***guardar los cambios... le dices que Yes*** y regla modificada y archivo de reglas actualizado





La última opción de ***Wizards*** es ***Online Update***, esto te permite parchear ***snort***, ***bajarte nuevas reglas***, hacer ***copias de seguridad*** de las mismas, etc... Lo dejo para tu propia investigación porque tampoco tiene mucho misterio.

Ahora pasemos a la siguiente ficha del menú... Vamos por la ***Ficha Logs***

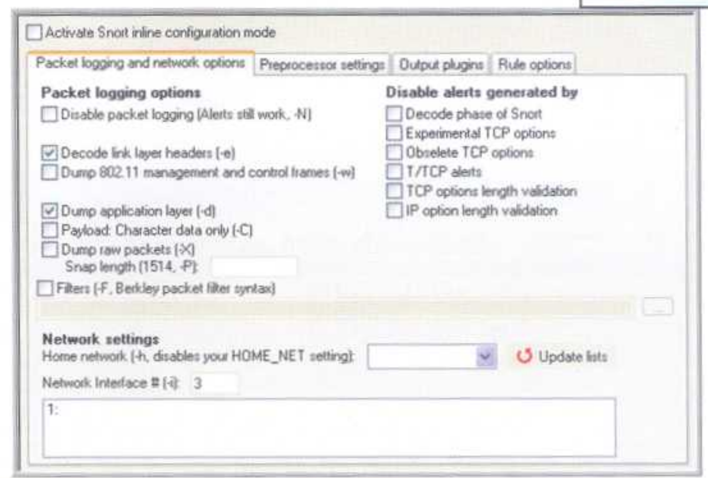
Sólo tiene **dos opciones** (***Options y log rotator***)



En ***options*** puedes configurar lo que serían las opciones de ***línea de comandos de snort***, es decir, ***aquello de -d -e -l -c, etc..***

Observa que tras la casilla de verificación suele haber una indicación de la equivalencia, por ejemplo ***Decode link layer*** equivaldría a teclear ***snort -e,*** y así con todas

Pantalla 52. Opciones de línea de comandos desde IDSCenter



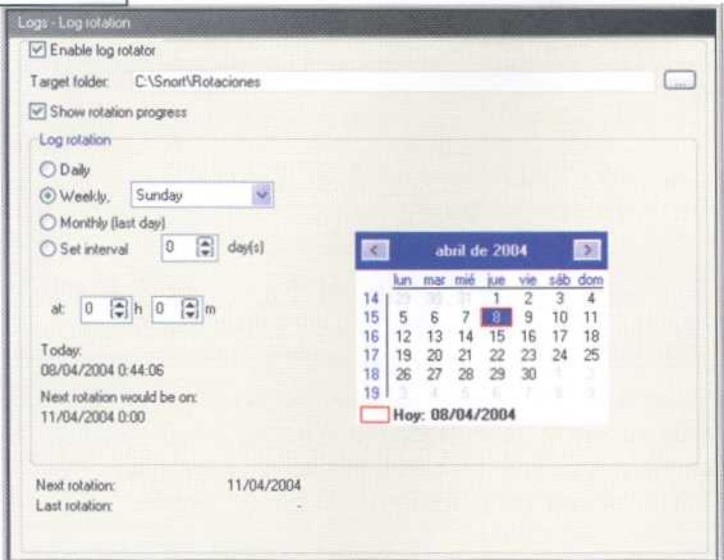
(Advierte que se seleccionó la tarjeta de red 1, si dispones de más de una y tienes a ***snort*** escuchando por otra deberás poner el número correspondiente, esto corresponde a la **opción -i de la línea de comandos)**

Y seguro que ya lo has notado.. Pero **hay 4 fichas en la pantalla central**, prueba tú mismo con ellas que son *"pocas y cobardes"*, *vamos que están chupadas a estas alturas*.

Para finalizar con ***Logs***, tenemos ***Log Rotation.***

Es útil porque **podemos decidir el tiempo de circulación y existencia de los archivos de registro** en el PC, por ejemplo podemos explicar a ***snort*** que guarde un histórico de alertas y *logs* semanalmente cada domingo en un directorio concreto del disco, que no debe ser el mismo de *logs*, es la única salvedad.

Pantalla 53. Rotación de logs



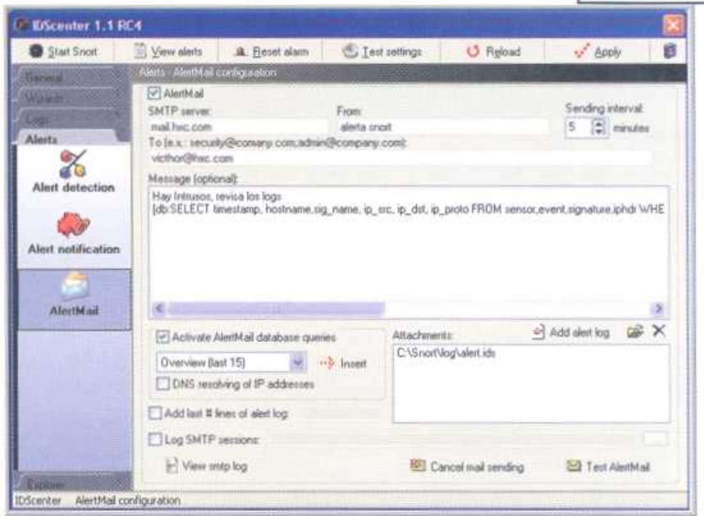
Nos quedan dos fichas... bueno una... por que ***la de Explorer se explica por sí misma***

La ficha de Alerts en la zona izquierda, tiene ***tres opciones: Detection, notification y Mail***

Estas opciones nos servirán para configurar ***IDSCenter*** y que emita sonidos cuando se produzcan alertas, o nos envíe un mail, ejecute un programa determinado, monitorizar el acceso a la Base de Datos, etc...

No nos detendremos en ellas, son fáciles de configurar y no debes tener problemas en ello, te pongo unos ejemplos.


Pantalla 54. Envío de alertas por mail

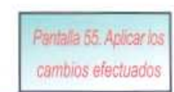


Seguro que esta última pantalla de resulta algo más costosa de entender... nada... no tiene misterios, ***enviaría un mail a la dirección indicada adjuntando el archivo alert.ids y con la consulta a la base de datos, es tan simple como pinchar en Insert***

Es mejor que **por el momento no uses al servicio de alertas** externas de esta ficha, normalmente esto se configura al final en entornos de producción y una vez que todo funciona como queremos se implementan, pero en *"fase de pruebas"* no es muy útil disponer de esas funciones hasta que todo ruede fino, fino.

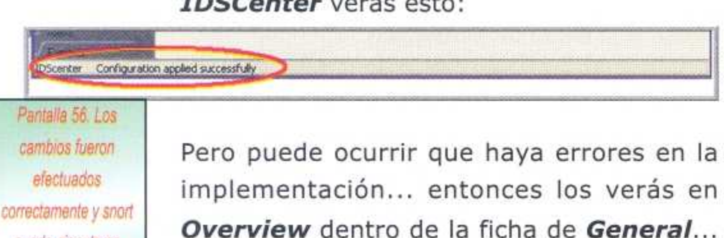
Así que para esta práctica las deshabilitaremos, más tarde ya harás tus ensayos...

Una vez que todo está terminado, ***pincharemos en Apply*** (parte superior de la ventana)


Pantalla 55. Aplicar los cambios efectuados

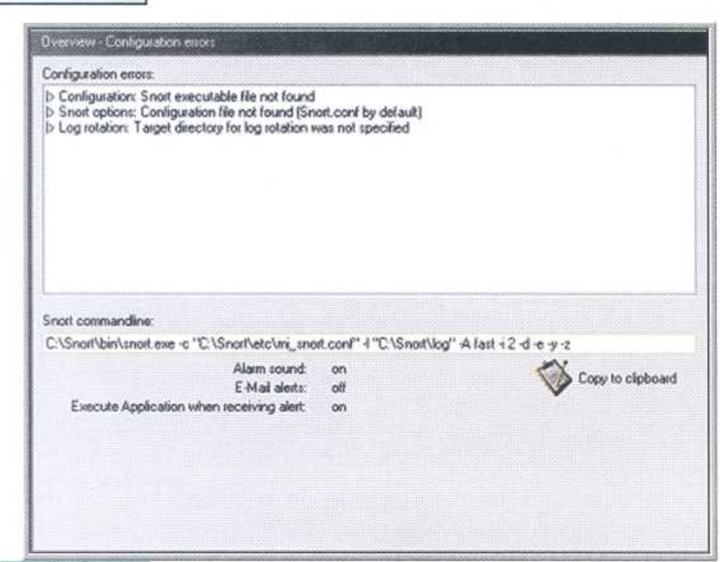


Y si todo fue bien **en la barra de estado** de ***IDSCenter*** verás esto:



Pantalla 56. Los cambios fueron efectuados correctamente y snort puede ejecutarse

Pero puede ocurrir que haya errores en la implementación... entonces los verás en ***Overview*** dentro de la ficha de ***General***...

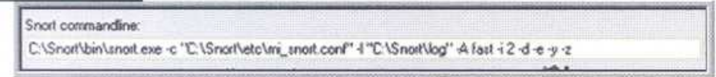


Pantalla 57. Se produjeron errores en los cambios. Hay que subsanarlos

Como ves en ***Configuration errors*** se muestran los errores encontrados, en este caso tres errores que habrá que subsanar... pero no será nuestro caso si has seguido punto por punto la configuración que se ha explicado.

También en la pantalla de ***Overview*** puedes ver cómo quedaría la ***línea de comandos de snort si la hubiésemos tecleado desde la shell...***

Pantalla 58. Línea de comandos que será ejecutada



Ahora sólo nos falta ponerlo en marcha... **desde la shell o desde** el mismo ***IDSCenter***...



Y aparecerá...



Lo que significa que ***snort*** está a la espera y rodando....

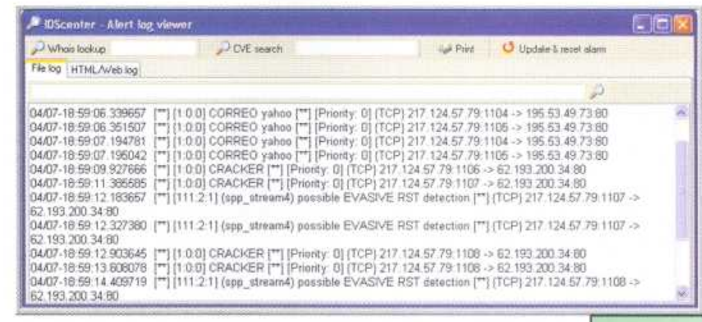
Pantalla 60. Ejecución de la consola de snort

Ahora navegamos por algunas páginas de sexo, por los foros de hackxcrack, etc... cualquiera de esas cosas que rompen nuestras reglas....

Luego vemos las ***alertas y logs pinchando en View Alerts***

Pantalla 61. Visor de Alertas desde IDSCenter





Pantalla 62. Detalle de las Alertas desde IDSCenter

Ya está todo, es bastante potente, ha sido muy rápido y tiene muchas opciones; pero confiamos en que las puedas resolver por ti mismo sin mayor dificultad... y si no.... nos tienes en los foros:

http://www.hackxcrack.com

Hasta la próxima

ALTO!! No te vayas, un momento... el mes pasado nos dijiste que íbamos a ***realizar pruebas de comportamiento*** del IDS e intentar evitarlo o provocar un ***DoS***... *¿donde está eso?*

Pues llegará en el próximo artículo, la idea inicial era 3 artículos para ***snort***, 1 para ***Firewalls*** y 1 de ***proxys***,.. así que cambiaremos la estructura.... quedará así:

Los tres artículos de ***snort*** que ya tenemos resueltos.

El próximo artículo: ***Escaneres*, *pruebas de comportamiento y estabilidad del sistema***. Una cuestión importante... es que esto no sólo servirá para analizar a ***snort***... servirá **para cualquier servidor, protegido o desprotegido, para una *firewall*, para probar un *router*, un pc cualquiera...**

Dejaremos para *"mas lejos" Firewalls y Proxys...*

Ahora sí



XBOX LIFE VII
CREANDO NUESTRO SLAYER
(POR ALFONSO MENKEL)

En este tema del curso vamos a empezar a utilizar PHP con la navegación web, vamos a aprender como enviar datos de una página a otra, a recogerlos y a tratarlos posteriormente.

Hol@ a todos y todas, llegamos al final del viaje, justo antes de las vacaciones. Este es el último artículo de XBOX LIVE. Ha sido un enorme placer haber escrito para la revista, Gracias.

Bueno, este mes vamos a crear un SLAYER. Habrá peña que se estará preguntando qué demonios es esto, pues muy sencillo, vamos ha crear un CD o DVD que nos instalará distintas aplicaciones.

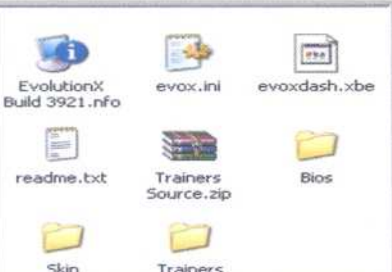
¿Por qué no expliqué esto antes en vez de liaros con el FTP, Boxplorer y demás? Pues muy sencillo, para enseñaros como funcionan las cosas, que es la filosofía de la revista y la cual comparto.

1.) Lo que necesitamos:

Xbox con mod chip instalado.
PC preparado.
Evolution x (ultima versión)
Evox Skinner (veremos la versión 1.2.7).
Programa de dibujo y de retoque fotográfico.

Todos los programas los podéis encontrar en la Web de la revista (www.hackxcrack.com), así que ya lo estáis descargando e instalamos el Evox Skinner.

Ahora creamos una carpeta llamada "Slayer" en nuestro ordenador y descomprimimos el Evolution X en ella. Veremos esto:



Borramos todo exceptuando:

evoxdash.xbe
evox.ini
Carpeta Skin (pero vacía).

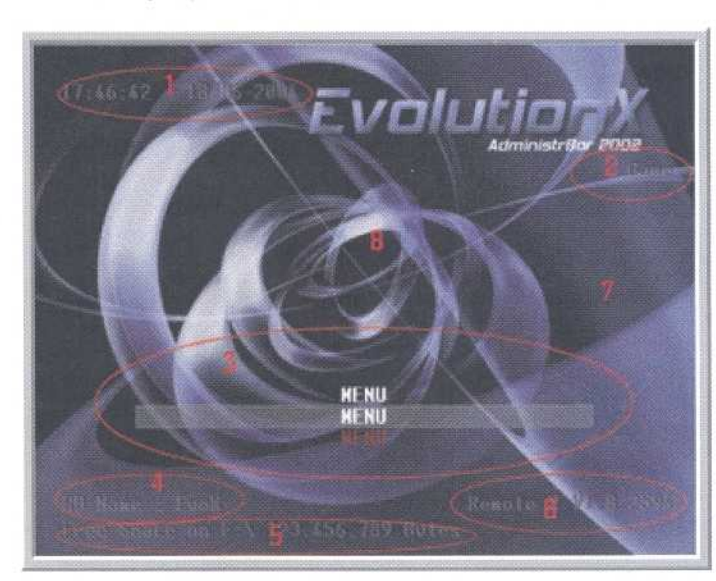
2.) Creando una Skin de Evolution X

Ahora crearemos una Skin para el veos. Para los que no sepan lo que es una skin, decirles que es la apariencia del evolution x, la imagen de fondo, la imagen de carga, los colores del menú.

Supongo que tendréis ya instalado el Evox Skinner y un programa de dibujo.

Antes de seguir debo aclarar unas cosillas sobre los skin de evolution x. En los Skins de evolution x hay varios "objetos" que podemos definir.

Veamos el ejemplo:



1.) Hora y fecha.
2.) Contenido del DVD
3.) El menú, donde podemos definir los colores del borde de la barra, de la barra, del texto que ha pasado o esta en la barra y el texto que esta fuera de la barra.
4.) Nombre puesto en el archivo evox.ini
5.) Espacio libre en F:
6.) Versión de la BIOS
7.) Imagen de fondo
8.) El logotipo de evolution X, que no aparece en la imagen porque esta en movimiento.

Esto es lo que vemos en un skin normal que viene con el evolution x. A excepción de la imagen de fondo y una posición del logotipo de evolution x, el resto no son obligatorios.

Ahora debemos ir al programa de dibujo y crear una imagen jpg de 640x480, ya que el propio programador dice que si la imagen es mayor o menor puede dar fallos.

Esta imagen puede ser como tú quieras, siempre y cuando no interfiera la lectura del menú.

Esta es la imagen que he creado para la ocasión. Tiene 4 campos: IP, Estatus del DVD, Espacio libre en E: y Espacio libre en F:



He puesto estos campos porque nos serán útiles a la hora de instalar dashboards y utilidades.

Debéis pensar en qué información necesitáis para instalar el contenido del cd.

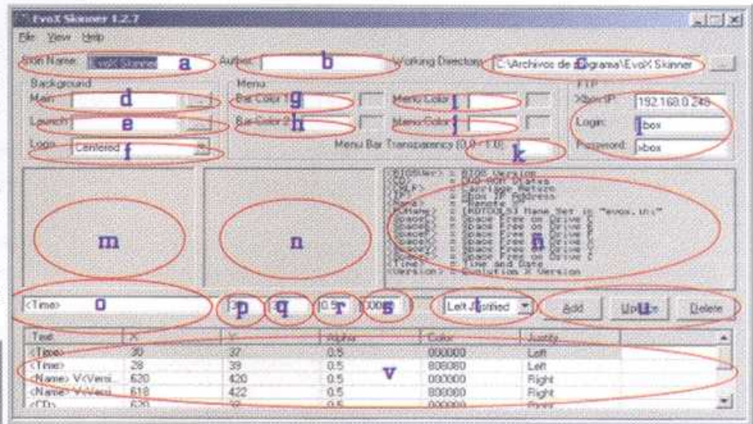
Después de crear nuestra primera imagen, creamos una segunda imagen del mismo tamaño, que será el que usaremos como fondo en la pantalla de carga.

Este es el mío:



Muy simple, pero el mensaje queda muy clarito.

Ahora arrancamos el evox Skinner.



Paso a explicar qué es cada campo y luego configuraremos nuestro skin:

a.) Nombre del Skin
b.) Autor del Skin (Nuestro nick o nombre).
c.) Directorio de trabajo para el skin.
d.) Imagen de fondo para el menú.
e.) Imagen de carga.
f.) Situación de logo de evolution X, tenemos que elegir una. Yo prefiero colocarlo en una esquina para que no se vea demasiado.
g.) Color del relleno de la barra de selección.
h.) El color del borde de la barra de selección.
i.) Color del texto que esta por encima o en la barra de selección.
j.) Color del texto que esta por debajo de la barra de selección.
k.) La transparencia de la barra de selección.
l.) Configuración FTP para pasar los skin creados directamente a la consola.
m.) Vista previa del fondo del menú.
n.) Vista previa de la imagen de carga.
ñ.) Etiquetas que podemos usar y lo que hacen. (Esto lo explicare en el ejemplo).
o.) Lugar donde se insertara la etiqueta.
p.) Distancia en pixels desde la esquina superior izquierda hacia la derecha de la imagen donde queremos que se vea la etiqueta.
q.) Distancia en pixels desde la esquina superior izquierda hacia abajo de la imagen donde queremos que se vea la etiqueta.
r.) La transparencia del texto de la etiqueta.
s.) Color del texto de la etiqueta.
t.) Justificación del texto de la etiqueta.
u.) Tres botones donde poder hacer distintas opciones. Add Anadir una etiqueta, Update Actualizar una etiqueta, Delete Borrar una etiqueta.
v.) Lista de las etiquetas insertadas, las podemos modificar u borrar con los botones de U.

Vemos un montón de cosas, pero no os asustéis esto es muy fácil.

La mejor forma de entender esto es con un ejemplo:

Lo primero que debemos hacer es seleccionar un directorio de trabajo. Vamos a la carpeta "Skin" que tenemos en el directorio "Slayer" antes creada y creamos una carpeta. En la opción C seleccionamos la carpeta que acabamos de crear.

En esta carpeta (yo la he llamado Slayer) copiamos las dos imágenes creadas anteriormente.

En la opción A, ponemos el nombre de nuestra skin, en mi caso Slayer (que poco original que soy).

En B, ponemos nuestro Nick o Nombre. En D seleccionamos la imagen que queremos de fondo. En E seleccionamos la imagen de carga.

Ahora en F seleccionamos donde queremos que esté el logotipo de Evolution X, yo lo he puesto en la esquina inferior derecha, casi no ser percibe.

G, seleccionamos el color del fondo de la barra. Seleccionar un color que no sea dañino a la vista y que quede bien con la imagen de fondo, evitar los colores como el amarillo que fastidia un montón la vista.

En H seleccionamos el color del borde de la barra de selección. I y J, cuando creo un skin le pongo a estos dos campos el mismo color.
K, si queremos que la barra tenga una transparencia poner un valor aquí, el 0.0 será totalmente transparente y el 1.0 será totalmente opaco.

Bueno, ahora llega lo entretenido, no es complicado pero si laborioso, la introducción de etiquetas.

Primero explicar que de la imagen que hemos creado no todo se va a ver, parte de la imagen se queda fuera, para saber lo que no se verá podemos pinchar en View->Preview.

En la ventana que nos sale veremos una vista previa de nuestro skin, pinchamos en view->Tv border y nos saldrá un cuadrado señalando el borde de la televisión, pues desde la esquina superior izquierda de ese cuadrado se contaran los píxel para situar las etiquetas.

Ahora ha llegado el momento de saber qué es eso de las etiquetas. Para los que han programado en HTML no les sonara tan raro; son unas palabras clave ente estos dos símbolos < > que tienen un significado para el evolution x.

En mi caso tengo cuatro campos donde colocar cuatro etiquetas, que son:

la IP de la Xbox
el estado del DVD
el espacio libre en E:
y el espacio libre en F:

Vosotros tendréis los vuestros y estarán situados en distinto sitio que el mío.

Borramos todas las etiquetas que vienen en V y guardamos.

Estos son los valores que he dado para mi skin:

Text	X	Y	Alpha	Color	Justify
<IP>	110	153	1.0	0066FF	Left
<CD>	110	217	1.0	0066FF	Left
<SpaceE>	389	153	1.0	0066FF	Left
<SpaceF>	389	217	1.0	0066FF	Left

Como podemos ver, hay que introducir las coordenadas donde queremos que aparezca la etiqueta.

La etiqueta <IP> esta situada a 110 pixels hacia la derecha desde la esquina superior izquierda y a 153 hacia abajo desde el mismo punto.



Esto lo debéis calcular vosotros mismos. Debemos que tener en cuenta lo que ocupa el texto real de la etiqueta, así que en la pantalla de vista previa, pinchamos en View->text y veremos el texto real y lo que ocupa.

Guardamos todos los cambios y cerramos el programa, que con esto ya hemos terminado.

3.) Modificando el Evolution X.

Ahora vamos a usar unas funciones poco conocidas del evolution x para que copie el contenido de una o varias carpetas al disco duro de nuestra consola.

Abrimos el exvo.ini en un editor de texto como el notepad.



La configuración...

La configuración del evolution x se explico en números anteriores de la revista, si no sabes de lo que estamos hablando pásate por la web de la revista (www.hackxcrack.com) y pide los números atrasados.

No es obligatorio pero es aconsejable que se configure el evolution x para poder acceder por FTP.

Nos vamos a la parte del fichero donde pone [MENU] y borramos todo lo que haya debajo de Section "Root" {.

Nos quedaría así:

```
[MENU]
Section "Root"
{

}
```

Ahora nos vamos al directorio Slayer y creamos 3 carpetas con estos nombres: C, E, F (Si no tenéis partición F no la tenéis que crear. Si tenéis partición G Deberíais crear una carpeta G).

Copiaremos el contenido de las carpetas a su unidad. Cuando digo el contenido de las carpetas quiero decir que si tenemos archivos sueltos en la carpeta C del PC, estos archivos se copiaran a C: raíz de la consola, no se copiaran en ninguna carpeta.

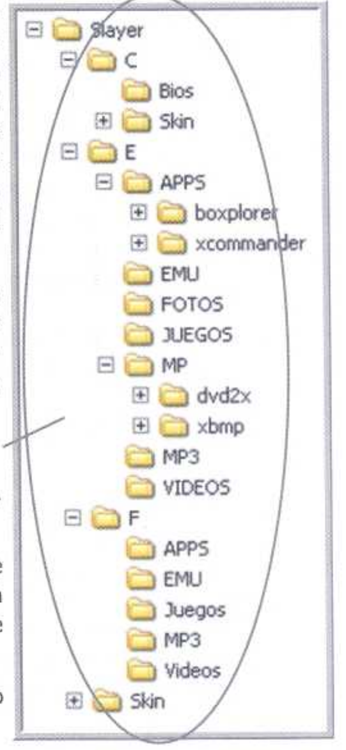
Si una carpeta que va a ser copiada no existe en la unidad de destino, se creará.

Si la carpeta a copiar existe, el contenido de esa carpeta se copiara a esa carpeta.

Parece un poco lioso pero no lo es tanto, lo veréis mejor con el ejemplo.

En las carpetas creadas en nuestro PC (C, E y F) pasaremos lo que queramos copiar al HD de la consola. Por ejemplo, para el evolution X tendríamos que copiar la carpetas Skin y los archivos evoxdash.xbe y evox.ini a la carpeta C; pero si quisiéramos copiar el Boxplorer a la unidad E, tendríamos que crear una carpeta Boxplorer en la carpeta E y dentro de el pasaríamos los ficheros del boxplorer.

Pero imaginad que queremos copiar el boxplorer en la carpeta de aplicaciones (apps) pues tendríamos que crear una carpeta APPS en la carpeta E y en APPS tendríamos que crear una carpeta llama Boxplorer y dentro de esta copiaríamos los archivos del boxplorer.



Pongamos el ejemplo de una consola con el chip recién instalado pero sin evolution x ni nada. Queremos instalar el evolution x, y algunos programas.

Este seria el árbol de directorios:

Aunque no se vean los archivos, están en sus carpetas.

Una vez tengamos la estructura de carpetas, nos vamos al evox.ini en el editor de texto (que no os he dicho que cerréis).

Esto es el código que he insertado ente las {}.

```
Item "Instalar Evox",@201
Item "Instalar Evox Con Particion F",@202
Item "Formatear F:\",@210
Item "Configuracion",ID_Settings
Item "Reiniciar",ID_Quick_Reboot
Item "Apagar",ID_Power_Off
```

Después de la "}" con una separación de 2 líneas en blanco, he introducido este condigo:

```
[Action_01]

Info "Se instalara evolution x y programas"
Warning "Deseas Continuar?"
Copy "\c\" "c:\"
Copy "\e\" "e:\"

[Action_02]

Info "Se instalara evolution x y programas con particion F"
Warning "Deseas Continuar?"
Copy "\c\" "c:\"
Copy "\f\" "e:\"
Copy "\e\" "f:\"

[Action_10]

Info "Format F"
Warning "This will format F"
Progress "Formatting F"
#
# This will format F:\
#
Format f:
```

Vamos a aclarar esto un poco:

En el artículo de evolution x ya explicamos que podíamos insertar ítems en el evolution x para crear los menús.

Item "Instalar Evox",@201

* **Item** : inserta un item.
* **"Instalar Evox"**: Texto que se vera en el menú.
* **,@201**: La acción que debe ser ejecutada si esta opción es la elegida.

Entonces queda claro que si pulsamos en la opción Instalar evox ejecutara la acción 01.

Ahora explico las acciones:

[Action_01]

Info "Se instalara evolution x y programas"
Warning "Deseas Continuar?"
Copy "\c\" "c:\"
Copy "\e\" "e:\"

* ***[Action_01]***: Número de la acción. En el item debe tener un 2 delante del número de acción.

* ***Info "Se instalara evolution x y programas"***: Titulo informativo sobre la acción a ejecutar.

* ***Warning "Deseas Continuar?"***: Aviso con posibilidad de cancelar o continuar.

* ***Copy "\c\" "c:\"*** y ***Copy "\e\" "e:\"***: Copia el contenido de tal carpeta a tal otra.

Entonces nuestro menú hará lo siguiente.

Si ejecutamos la acción "Instalar Evox" copiaremos el contenido de la carpeta C a la partición C: y el contenido de la carpeta E a la partición E:

En cambio, si elegimos la opción "Instalar Evox Con Particion F", lo que haremos será copiar el contenido de la carpeta C a C:, la carpeta F a la partición E: y la E a la partición F:.

Os podéis estar preguntando ¿Por qué lo hago así? ¿No seria más lógico copiar la carpeta F a F: y seguir como estaba en la primera acción?

Pues vamos por partes. Si tenemos el disco duro que viene con la consola y tiene partición F:, este será de dos GB y prácticamente ningún juego cabe en esta partición. En cambio, en E tenemos el espacio suficiente para copiar prácticamente cualquier juego (exceptuando varios juegos que ocupan seis GB o mas).

Entonces pensé que lo mejor seria usar los dos GB para programas y dejar E: libre para los juegos. Por eso en la carpeta F solo están los directorios y no los archivos, para así solo copiar la estructura de directorios por si algún día se decidiese usar estas carpetas.

Los demás ítems fueron explicados en anteriores números.

Dejamos dos líneas en blanco justo después de la última acción.

Nos vamos a la parte de configuración (arriba del todo).

SkinName = Original

Donde pone Original debemos poner el nombre de nuestra Skin.

Ejemplo:

SkinName = Slayer

Guardamos y cerramos.

El contenido de la carpeta Slayer la quemamos en un CD-RW o DVD como os enseñe en anteriores números.

Lo insertamos en la consola y a instalar.



Recordad que los programas que vayáis a instalar deben estar debidamente configurados.

Este skin lo podéis bajar de la Web de la revista.

Bueno pues ahora sí que hemos acabado el viaje. Aún quedan muchas cosas sobre Xbox que no he explicado, pero son más del tipo "BRICO-XBOX" y no creo que este sea el lugar apropiado donde publicar este tipo de artículos.

Quisiera dar las gracias otra vez a todos los que me han leído, apoyado y sufrido durante estos 7 meses.

PD. Hasta que nos volvamos a ver.
Salu2, Bye.